(2)

NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

A STUDY OF THE THERMAL PROFILES
DURING AUTOGENOUS ARC WELDING

by

Robert L. Ule

March 1989

Thesis Advisor Yogendra Joshi

REPORT DOCUMENTATION PAGE

1a Report Security Classification: Unclassified	1b Restrictive Markings
2a Security Classification Authority	3 Distribution Availability of Report: Approved for public release: distribution is unlimited.
2b Declassification Downgrading Schedule	
4 Performing Organization Report Number(s)	5 Monitoring Organization Report Number(s)
6a Name of Performing Organization: Naval Postgraduate School; 6b Office Symbol (if applicable): 34	7a Name of Monitoring Organization: Naval Postgraduate School
6c Address (city, state, and ZIP code): Monterey, CA 93943-5000	7b Address (city, state, and ZIP code): Monterey, CA 93943-5000
8a Name of Funding Sponsoring Organization; 8b Office Symbol (if applicable)	9 Procurement Instrument Identification Number
8c Address (city, state, and ZIP code)	10 Source of Funding Numbers: Program Element No; Project No; Task No; Work Unit Accession No

11 Title (include security classification): A STUDY OF THE THERMAL PROFILES DURING AUTOGENOUS ARC WELDING

12 Personal Author(s): Robert L. Ule

13a Type of Report	13b Time Covered From To	14 Date of Report (year, month, day)	15 Page Count
Master's Thesis		March 1989	163

16 Supplementary Notation: The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 Cosati Codes			18 Subject Terms (continue on reverse if necessary and identify by block number)
Field	Group	Subgroup	Welding, Computer, Modeling, Thermography, Heat Transfer, arc

19 Abstract (continue on reverse if necessary and identify by block number)

The three dimensional transient temperature variations during autogenous Gas Tungsten Arc Welding are determined. The model employs a combination of un-equally spaced moving meshes to minimize the total number of nodes. Finite differencing is used for the spatial terms. The resulting ordinary differential equations for the transient evolution of thermal transport are solved using the fourth order Runge-Kutta technique. The temperature dependent thermal properties and latent heats of phase transformations are accounted for. Computations are carried out for a rectangular parallelepiped, with convective and radiative surface thermal conditions. Results are presented for the evolution of thermal profiles during ideal welding conditions. These are next compared with variations obtained due to defects such as weld track misalignment and inclusions. A study of the startup and shutdown transients makes it possible to control the cooling rate during these transients. The potential use of this model in the development of an expert welding system using infrared imagery is indicated. In addition, a low cost infrared detector using an indium-arsenide diode is prototyped to determine its feasibility for production welding control.

20 Distribution Availability of Abstract	21 Abstract Security Classification	
☒ unclassified unlimited ☐ same as report ☐ DTIC users	Unclassified	
22a Name of Responsible Individual: Yogendra Joshi	22b Telephone (include Area code): (408) 646-3400	22c Office Symbol: 69Ji

DD FORM 1473,84 MAR — 83 APR edition may be used until exhausted; All other editions are obsolete — security classification of this page

Approved for public release: distribution is unlimited.

A Study of the Thermal Profiles
During Autogenous Arc Welding

by

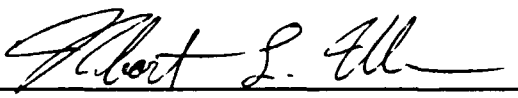
Robert L. Ule
Lieutenant Commander, United States Navy
B.S., Harvey Mudd College, 1977

Submitted in partial fulfillment of the
requirements for the degrees of

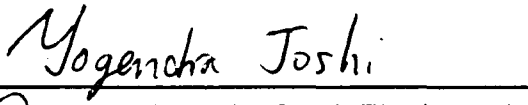
MASTER OF SCIENCE IN MECHANICAL ENGINEERING
and
MECHANICAL ENGINEER

from the

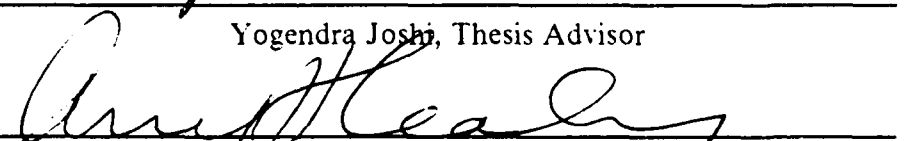
NAVAL POSTGRADUATE SCHOOL
March 1989

Author: 

Robert L. Ule

Approved by: 

Yogendra Joshi, Thesis Advisor



Anthony J. Healey, Chairman,
Department of Mechanical Engineering

Gordon E. Schacher,
Dean of Science and Engineering

ABSTRACT

The three dimensional transient temperature variations during autogenous Gas Tungsten Arc Welding are determined. The model employs a combination of un-equally spaced moving meshes to minimize the total number of nodes. Finite differencing is used for the spatial terms. The resulting ordinary differential equations for the transient evolution of thermal transport are solved using the fourth order Runge-Kutta technique. The temperature dependent thermal properties and latent heats of phase transformations are accounted for. Computations are carried out for a rectangular parallelepiped, with convective and radiative surface thermal conditions. Results are presented for the evolution of thermal profiles during ideal welding conditions. These are next compared with variations obtained due to defects such as weld track misalignment and inclusions. A study of the startup and shutdown transients makes it possible to control the cooling rate during these transients. The potential use of this model in the development of an expert welding system using infrared imagery is indicated. In addition, a low cost infrared detector using an indium-arsenide diode is prototyped to determine its feasibility for production welding control.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

SYMBOL	DESCRIPTION	UNITS
A_{xy}	Area of the x-y plane	mm^2
α	Thermal diffusivity	m^2/s
Bi	Biot number	*unity*
C_v	Volumetric heat capacity	$J/m^3\,^\circ K$
ε	Surface emmisivity	*unity*
Δq	Energy added over a finite time step	J
Δt	Finite time step	s
Fo	Fourier number	*unity*
H	Volumetric specific enthalpy	J/m^3
h	Convective heat transfer coefficient	$W/m^2\,^\circ K$
J	Flaw Correlation Coefficient	mm
k	Thermal conductivity	$W/m\,^\circ K$
n	Non-dimensional heat input rate operating parameter	*unity*
Q	Arc Power	W
ρ	Density	kg/m^3
σ	Stefan-Boltzmann constant	$W/m^2\,^\circ K^4$
T	Temperature	$^\circ K$
T_∞	Background temperature	$^\circ K$
T_{qss}	The quassi-steadystate temperature field about the arc	$^\circ K$
θ_i	The initial plate temperature	$^\circ K$
θ_r	A reference temperature at which a welding parameter is to be determined	$^\circ K$
v	The velocity of the arc	m/s

I. INTRODUCTION

The Navy's David Taylor Research Laboratory is currently in the process of developing an expert welding system. This integrated automatic welding system will be used for the Gas Metal Arc (GMA) welding of submarine hulls. This technology is mandated by the problems in mass production welding with the new high strength, low alloy steels. These steels have very tight limits on cooling rates which result in very low metal deposition rates. To meet production requirements, the use of an expert welding system is deemed mandatory. One portion of the expert welder is the infrared vision system. To effectively use this system it is desired to determine how various welding parameters affect the thermal signature at the surface of the plate being welded. However, no numeric models have been developed to supplement these experiments. Due to the cost and time associated with laboratory experiments, a numeric model of the welding process was needed to aid in this development.

Three specific problems were identified as candidates for investigation of their effect on the surface temperature profile. These were:

- The effect of inclusions in the weld.
- The effect of weld seam-to-arc misalignment
- The effect of start-up and shut-down on cooling rates.

The first two of these, by definition, are unsymmetrical problems. This precludes the use of symmetry in the computer model, something most models use to control the number of nodes to solve. In general, the solution cost is at best based on the square of the number of nodes for any solution technique requiring matrix inversion. However, since accurate results are desired, a three dimensional, variable coefficient model is the requirement.

The use of the finite element method for three dimensional weld modeling has been under development for a number of years. The current finite element models have included features such as temperature dependant material properties, complex and changing geometries and transient heat conduction analysis. The foremost of these is Goldak [Ref. 1] who concludes that present day computers are not capable of solving meaningful three dimensional problems. He estimates that it would take a CRAY-2 computer about 6 hours to solve a 10 second problem of interest using the finite element method with

only 659 nodes. This is because the sparse matrix resulting from the formulation must be solved using an iterative process such as Gauss-Seidel. Two researchers, Mangonon and Mahimkar [Ref. 2], have successfully developed a simple three dimensional model. When using 2500 nodes they took four hours of CPU time to simulate six seconds of real time. They used a grid spacing of 2.12 millimeters, which allowed them to take larger time steps, instead of the 1.0 millimeter grid spacing recommended by Goldak. To reduce the number of nodes, most researchers have invoked symmetry whenever it was possible. The end result of these difficulties is that welding models have been limited in scope. Most are limited to predictions of the weld pool geometry and the cooling rate in the heat affected zone, since these are easy to obtain.

These computational limitations, compounded by the need of a model that does not use symmetry, forced a reconsideration of the modeling technique to use. The explicit finite difference technique is the simplest and most direct of all of the numeric techniques for solving the heat conduction equation. Since the equations are not coupled, no matrix inversion is required to solve the system, greatly reducing the computational effort associated with large numbers of nodes. However, there remains the classic limitation on the time step of $Fo \leq \frac{1}{6}$. Where Fo is the Fourier number, and the 6 is for a three dimensional model. Of course, the Fourier number is not the only limitation on time step. The degree of accuracy is also dependant on the time step. Because of the large temperature gradients in the vicinity of the arc, very small time steps are required for accuracy. Unlike most time dependant problems, this condition is not transient, and dynamically changing the time step is not an option. Testing with a simple two dimensional model showed that for the temperature gradients of interest, the time step required to be within one percent of the time step independent solution was more limiting than the Fourier number limitation. Further numeric testing showed that the equivalent implicit finite difference model was just as accurate (or inaccurate) as the explicit finite difference model under these conditions, while having the more expensive cost of matrix solving.

With this insight into the unique problems of welding analysis, that accuracy controlled the time step, not the stability limit of the Fourier number, the explicit technique becomes a viable alternative. To allow taking as large a time step as possible and still have reasonable accuracy, the finite difference equations are written in the semi-discrete form. This allowed applying fourth order Runge-Kutta. The combination of the explicit technique and Runge-Kutta proved to be the key to the solution. This resulted in a very

fast, accurate three dimensional model of the arc welding process which was able to handle grids with large numbers of nodes.

Using these techniques the model was written and debugged. A series of test were conducted to compare the results of the model with the available analytical solutions and experimental data. Following the verification, each of the major goals was investigated. In addition, some simple laboratory experiments were designed to verify the specific results of the model. The model proved to be very versatile in answering the posed questions and is a valuable tool for continued research.

In the course of this research, a low cost alternative to the infrared camera system was proposed. In lieu of the expensive and delicate vision system, the use of a simple infrared diode was considered. There exists a commercially available indium arsenide diode which has an acceptable detection range. Experimental work was performed to determine if the diode had the sensitivity and accuracy necessary to be a useful industrial detector. The results of this testing, especially in light of the results of studying flaws and arc-to-seam misalignments, indicates that a simplified vision system would be a cost effective option.

II. THE THREE DIMENSIONAL WELD MODEL

A. SELECTION OF MODELING TECHNIQUE

Both the implicit and explicit finite difference techniques were considered. The benefits of the explicit technique are many. The code is easy to write for the three dimensional case since node ordering is not important. The memory storage requirements are substantially less since an N by N matrix is not being created. The solution cost is always N calculations per time step. The formulation can be developed in the semi-discrete form, where only the spatial derivatives are written using difference equations. The time derivative is then evaluated using fourth order Runge-Kutta. For a rectangular steel plate, the boundary conditions are simple to encode. Another possible technique, the implicit method is not limited by stability in choosing a maximum allowable time step based on the Fourier number. This time step limitation is discussed in detail in [Ref. 3].The benefit of the finite element method is that it easily adapted to the weld pool geometry and it is simpler to increase the nodal density in the vicinity of the arc.

Because of the simplicity of the explicit finite difference technique, a series of tests were performed to compare the implicit technique to the explicit technique for the problem of interest. These tests compared both methods, solving a simple problem with an analytical solution. The methods were then applied to a problem of a step transient. It was found that with the large temperature gradient of a step, the limitation on the Fourier number was not what limited the time step size. For a fixed grid size, it was found that both the implicit and explicit methods developed a greater than three percent error when the time step was increased to the Fourier number critical time step. The error is measured relative to a time step independent solution, which is obtained by halving the time step until the solution changes by less than 0.1 % . This solution is used in lieu of an analytical solution to calculate the error. In addition, the explicit method was always the more accurate of the two methods when solving problems with large temperature gradients. Thus no benefit would be gained by using the implicit technique.

To keep the number of nodes down to a solvable number but still have a fine grid mesh in the region of large temperature gradients, a combination of three mesh sizes is used. These meshes shown in Figure 1 include the fine mesh, medium mesh and coarse mesh. Each small rectangular region corresponds to a control volume with the

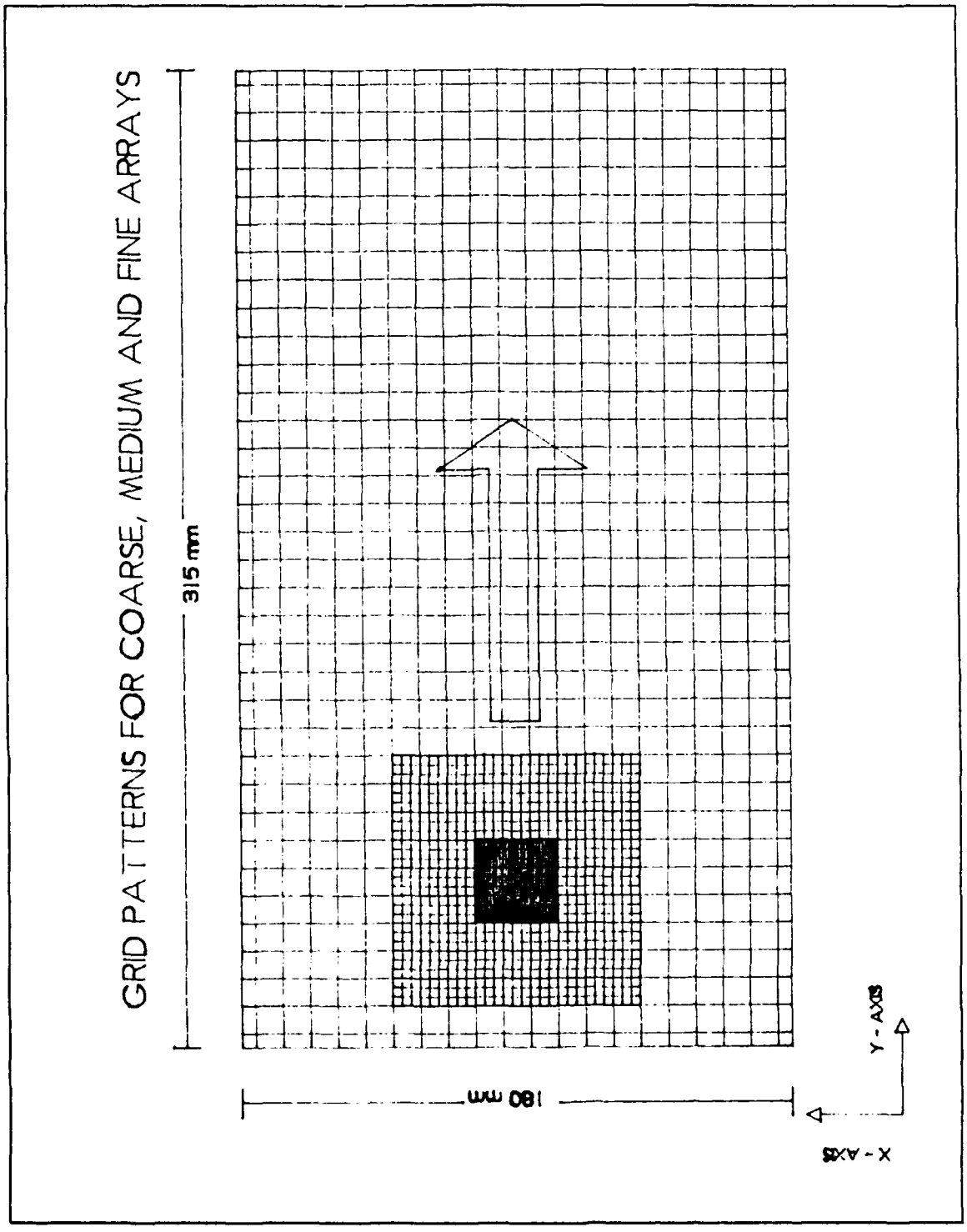


Figure 1. The Coarse, Medium and Fine Meshes

associated node located at the centroid. For the case of surface nodes, the node is actually at the surface and not at the centroid of the control volume; for the corner nodes of surfaces, the node is at the corner. This is consistent with the normal method of assigning nodes for finite differences. In the figure, this is shown by the difference in control volume size, since for a given mesh, the nodal spacing is constant.

The spacing of the fine mesh is one millimeter, the medium mesh is three millimeters and the coarse mesh is nine millimeters. The factor of three is chosen to allow smoother shifting between different sized meshes. The fine mesh moves with the arc through the medium mesh in three millimeter jumps. After the arc moves nine millimeters, the medium mesh shifts nine millimeters in the coarse mesh and the fine mesh is back at its initial point with respect to the medium mesh. In this way the arc is always located in the fine mesh grid while the size of the fine mesh grid is kept to a minimum.

To allow taking as large a time step as possible and still have good accuracy, the finite difference equations are written in the semi-discrete form. This allowed applying fourth order Runge-Kutta. Not only does this allow taking time steps up to the Fourier number limit, but it also stabilizes the system of equations and it is possible to take time steps in excess of the Fourier number limit and still have reasonable accuracy. For a Fourier number limitation of eight milliseconds, a time step of ten milliseconds is routinely used with an error of less that one percent. This error is based on taking smaller time steps until the solution converged. As before, this converged solution is the reference for the error.

B. CREATING THE MESHES

The geometry selected is a thick plate of steel. The selected size to model is 1″ by 7″ by 12″ steel. The model actually uses 27mm by 180mm by 315mm. The whole plate is modeled by the coarse zone using a two dimensional grid 21 by 36. The medium grid is centered about the arc and is 27mm by 81mm by 81mm for a grid of 10 by 27 by 27. The fine grid is centered about the arc, but does not penetrate the plate, and covers a size of 7.5mm by 27mm by 27mm for a grid of 8 by 27 by 27. This gave a total of 13,878 nodes. The coarse zone is made two dimensional since it is far enough away from the arc for the temperature profile to be essentially isothermal in the z (vertical) direction.

Pointers are used to keep track of the relative positions of the grids. When the grids are moved relative to one another the temperature of the finer grid nodes being left behind are averaged to provide a temperature to the coarser node which would occupy the same space. The finer grid nodes moving to where the coarser grid had been are all

assigned the temperature of the coarser grid point whose space they now occupied. The nodes are shifted only in increments of three of the finer grid, which corresponds to one increment of the coarser grid.

The problem of temperature dependant material properties, including phase transformation, is also treated. The material properties are constant in both the coarse and medium mesh zones and are evaluated at room temperature. In the fine zone, the change in enthalpy is calculated instead of the change in temperature. The corresponding temperature for a given enthalpy is found using a piece-wise continuous function. This took into account the austenite to ferrite transitions as well as melting. The temperature dependence of thermal conductivity is also similarly handled. The sizing of the fine zone insured that the high temperatures where these effects dominated are inside the fine zone mesh. In addition, a radiation boundary condition is added in the fine zone.

The use of different sized grids creates an additional problem of calculating the heat transfer at the interface zones. The bookkeeping becomes involved and a detailed explanation of how the program is written and the source listing are included in appendix A. A brief discussion of how the equations are set up on the mesh follows.

C. FINITE DIFFERENCE DERIVATION

1. The Fine Mesh Zone

The fine mesh region uses the enthalpy form of the finite difference conduction heat equation. In addition, the thermal conductivity is a function of temperature and so will be different at each node. The thermal conductivity used for each difference is the harmonic mean for the two nodes being differenced. A side view of the fine mesh and its interface with the medium mesh is seen in Figure 2. Some of the representative energy balance equations are listed next.

a. Interior Nodes

The most general case is an interior node, given in equation 2.1.

$$\Delta H_{i,j,k}^{n+1} = \left(\frac{2\Delta t K_{i,j,k}}{\Delta x^2} \right) \left[\frac{K_{i+1,j,k}(T_{i+1,j,k}^n - T_{i,j,k}^n)}{(K_{i+1,j,k} + K_{i,j,k})} + \frac{K_{i-1,j,k}(T_{i-1,j,k}^n - T_{i,j,k}^n)}{(K_{i-1,j,k} + K_{i,j,k})} \right.$$
$$+ \frac{K_{i,j+1,k}(T_{i,j+1,k}^n - T_{i,j,k}^n)}{(K_{i,j+1,k} + K_{i,j,k})} + \frac{K_{i,j-1,k}(T_{i,j-1,k}^n - T_{i,j,k}^n)}{(K_{i,j-1,k} + K_{i,j,k})} \qquad (2.1)$$
$$\left. + \frac{K_{i,j,k+1}(T_{i,j,k+1}^n - T_{i,j,k}^n)}{(K_{i,j,k+1} + K_{i,j,k})} + \frac{K_{i,j,k-1}(T_{i,j,k-1}^n - T_{i,j,k}^n)}{(K_{i,j,k-1} + K_{i,j,k})} \right]$$

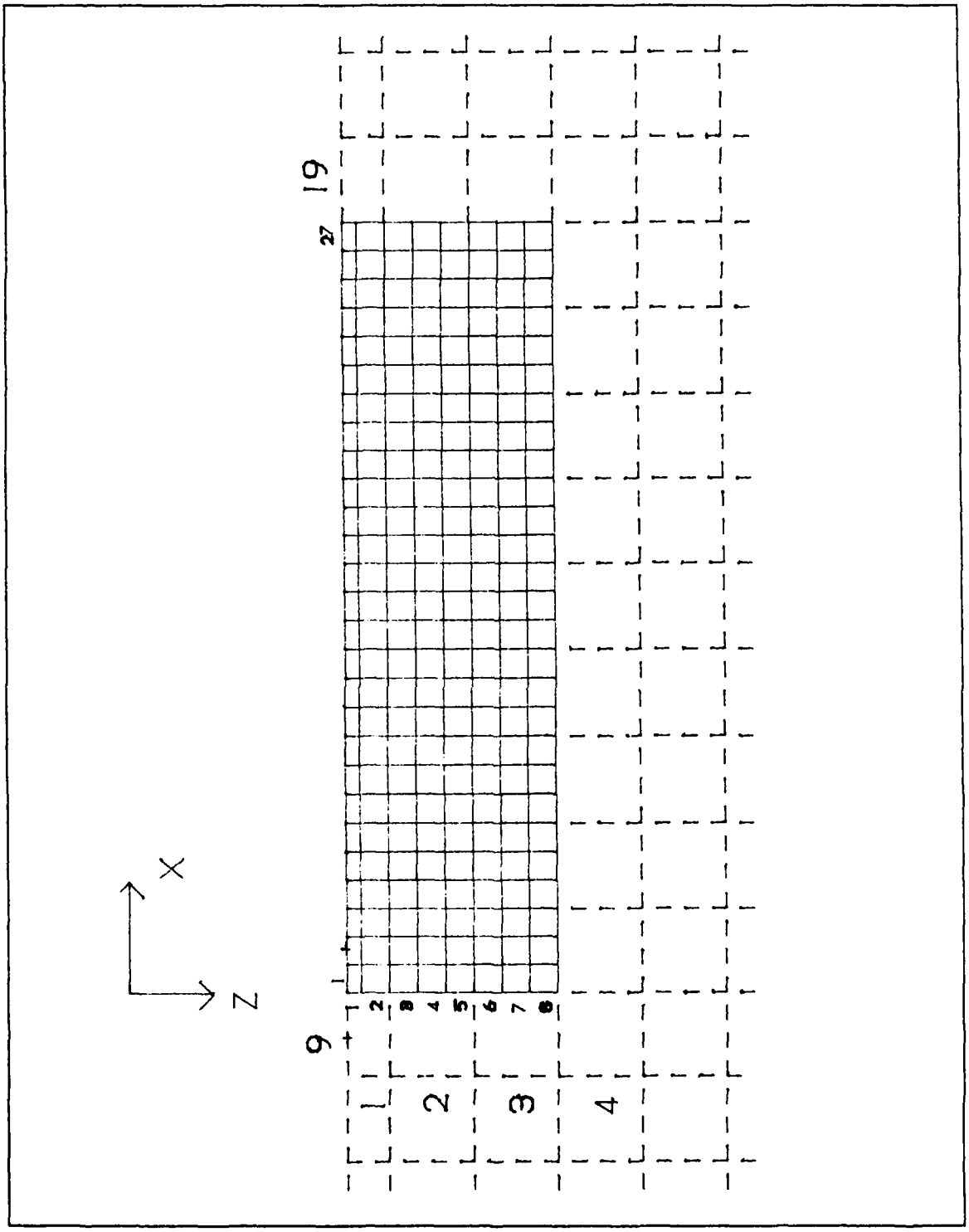


Figure 2. Side View of Fine and Medium Interface

For programming, equation 2.1 can be simplified by the function defined in equation 2.2. This calculates the harmonic mean of the thermal conductivity at each nodal pair.

$$G(T_1, T_2) = \frac{2(T_1 - T_2)}{\frac{1}{K(T_1)} + \frac{1}{K(T_2)}} \tag{2.2}$$

Combining with equation 2.1 yields

$$\begin{aligned} \Delta H^{n+1}_{i,j,k} = \left(\frac{\Delta t}{\Delta x^2} \right) \Big[& G(T^n_{i,j,k}, T^n_{i+1,j,k}) + G(T^n_{i,j,k}, T^n_{i-1,j,k}) + G(T^n_{i,j,k}, T^n_{i,j+1,k}) \\ & + G(T^n_{i,j,k}, T^n_{i,j-1,k}) + G(T^n_{i,j,k}, T^n_{i,j,k+1}) + G(T^n_{i,j,k}, T^n_{i,j,k-1}) \Big] \end{aligned} \tag{2.3}$$

b. Surface Nodes

For surface nodes there are three terms that contribute to the change in enthalpy. As shown in equation 2.4 these are the conduction, convection and radiation.

$$\Delta H = \Delta q_{cond} + \Delta q_{conv} + \Delta q_{rad} \tag{2.4}$$

The convection and radiation terms are given by.

$$\Delta q_{conv_{i,j,k}} = \frac{h}{\Delta z} (T_{inf} - T_{i,j,k}) \tag{2.5}$$

$$\Delta q_{rad_{i,j,k}} = \frac{C\,\sigma}{\Delta z} (T^4_{inf} - T^4_{i,j,k}) \tag{2.6}$$

For the surface nodes, the node immediately below the surface has twice the effect of the nodes on the side, because it has twice the heat transfer area. Thus q_{cond} must be modified accordingly.

2. The Medium Zone Mesh.

In the medium zone, all of the material properties are held constant. In this case it is simpler to write the equations in the temperature form of the finite difference equations. The material properties are evaluated at 300 K. The relationship of the medium mesh to the coarse mesh is shown in Figure 3.

a. Interior Nodes

Equation 2.7 shows the typical form for an interior node, where Fo is the Fourier number.

$$\Delta T^{n+1}_{i,j,k} = Fo[T^n_{i+1,j,k} + T^n_{i-1,j,k} + T^n_{ij+1,k} + T^n_{ij-1,k} + T^n_{ij,k+1} + T^n_{ij,k-1} - 6T^n_{i,j,k}] \tag{2.7}$$

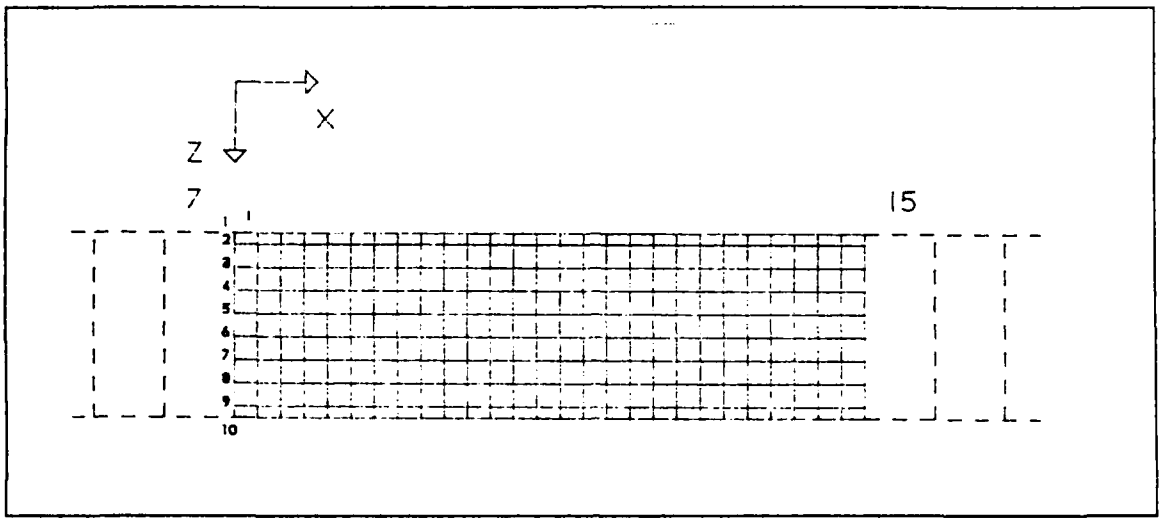


Figure 3. Side View of Medium and Coarse Interface

The medium zone is one of the more complicated to program since it interfaces with both the fine and the coarse grids. Interfacing problems are discussed in "4. The Interfaces" on page 11 For the surface nodes an additional term is added to take into account the free convection. Care is taken to note that the surface nodes have only one-half of the volume of an interior node. Equation 2.8 shows a typical form for a surface node,

$$\Delta T_{i,j,k}^{n+1} = Fo[T_{i+1,j,k}^{n} + T_{i-1,j,k}^{n} + T_{i,j+1,k}^{n} + T_{i,j-1,k}^{n} + 2T_{i,j,k+1}^{n} + 2BiT_{inf} - (6 + 2Bi)T_{i,j,k}^{n}] \quad (2.8)$$

where Bi is the Biot number for free convection.

3. The Coarse Zone Mesh

In the coarse zone, all of the material properties are constant. Since this models the plate far from the arc, it is only two dimensional in the X-Y plane. These equations are also written in the temperature form of the finite difference equations. The top and bottom surface are free convection while the ends are considered adiabatic, since the plate size for the model is based on selecting a large enough area for this to be true. Equation 2.9 shows the typical form for a coarse mesh node.

$$\Delta T_{i,j}^{n+1} = Fo[T_{i+1,j}^{n} + T_{i-1,j}^{n} + T_{i,j+1}^{n} + T_{i,j-1}^{n} + 2BiT_{inf} - (4 + 2Bi)T_{i,j}^{n}] \quad (2.9)$$

Care is taken with the top and bottom Biot numbers since the heat transfer area at a node's surface is one third that of the node's sides.

4. The Interfaces

The interfaces are handled by treating them as boundaries. Each grid mesh is solved separately, using the temperature from the boundaries of adjacent meshes. Pointers are used to keep track of the relative position of the fine mesh to the medium mesh and the medium mesh to the coarse mesh, since these move. Because the control volumes for the nodes are different at the interface, the finite difference equation is modified. A one dimensional example is shown to demonstrate this. For the case where nodal point m-1 is twice as far from nodal point m as nodal point m+1, the second derivative may be expressed by:

$$\left.\frac{\partial^2 T}{\partial x^2}\right|_m \approx \frac{\partial T/\partial x|_{m+1/2} - \partial T/\partial x|_{m-1/2}}{\Delta x}$$

These first derivatives may be in turn expressed as:

$$\left.\frac{\partial T}{\partial x}\right|_{m+1/2} \approx \frac{T_{m+1} - T_m}{\Delta x}$$

$$\left.\frac{\partial T}{\partial x}\right|_{m-1/2} \approx \frac{T_m - T_{m-1}}{2\Delta x}$$

These combine to form the following approximation to the second derivative at nodal point m.

$$\left.\frac{\partial^2 T}{\partial x^2}\right|_m \approx \frac{T_{m+1} + 0.5T_{m-1} - 1.5T_m}{(\Delta x)^2}$$

a. The Fine Zone to Medium Zone Interface

The details of the medium to fine interface are shown in Figure 2 and Figure 4. The fine to medium zone interfaces have two complicating features. The first is that the fine zone uses variable thermal properties while the medium zone uses constant. To insure conservation of energy, the constant thermal properties of the medium zone are used when calculating the change in enthalpy at the interface. This is done because using the same thermal properties insures that the amount of heat entering the medium zone is equal to the amount of heat leaving the fine zone. The second complication is

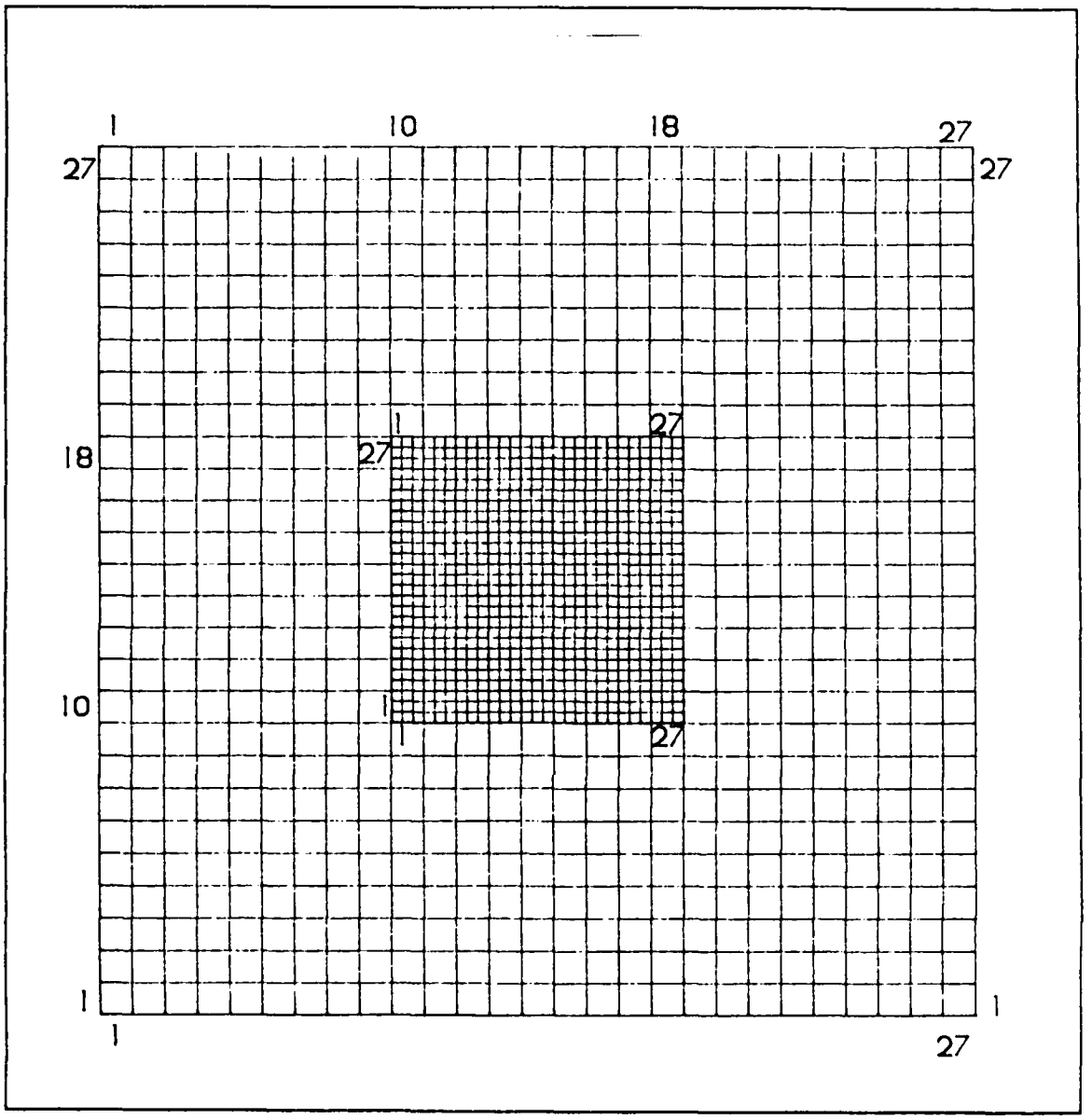


Figure 4. The Interface Between the Medium and Fine Meshes

the difference in node control volumes. This is seen in the last term of Equation 2.10, the interface relation for a non-surface, side node in the '-k' direction.

$$\Delta H_{i,j,k}^{n+1} = \frac{\Delta t}{\Delta x^2}\Big[G(T_{i,j,k}^{n}, T_{i+1,j,k}^{n}) + G(T_{i,j,k}^{n}, T_{i-1,j,k}^{n}) + G(T_{i,j,k}^{n}, T_{i,j+1,k}^{n}) + G(T_{i,j,k}^{n}, T_{i,j-1,k}^{n}) + G(T_{i,j,k}^{n}, T_{i,j,k+1}^{n}) + \frac{1}{2} K_{medium}(T_{medium}^{n} - T_{i,j,k}^{n})\Big] \quad (2.10)$$

The factor of 1.2 is due to the medium node being twice as far away as a fine node, with the change in enthalpy being related linearly to the diffusion distance. The heat transfer area for a medium node to one fine node is the same as a fine node to one fine node, so it does not effect the result. Figure 2 shows these relationships.

b. The Medium Zone to the Fine Zone

Interfacing the medium grid to the fine grid is a two step process. First, for every node of the medium interface, there are nine fine nodes at the interface. This is handled by taking the average of the nine nodal temperatures. This average temperature is then considered to be at a pseudo nodal point, which is only 2/3 the distance away as a normal medium nodal point. The energy balance equation then becomes,

$$\begin{aligned}\Delta T_{i,j,k}^{n+1} = Fo[&T_{i+1,j,k}^{n} + T_{i-1,j,k}^{n} + T_{i,j+1,k}^{n} + T_{i,j-1,k}^{n} + T_{i,j,k-1}^{n} \\ &+ 3/2\, T_{fine}^{n} - (5 + 3/2)T_{i,j,k}^{n}]\end{aligned} \tag{2.11}$$

As can be seen in equation 2.11, this adds a factor of 3/2 to the equation. A quick calculation will show that energy is conserved; the heat flux calculated by the fine mesh is equal to the heat flux calculated by the medium mesh.

c. The Medium Zone to the Coarse Zone

The interface of the medium mesh to the coarse mesh has identical geometry as going from the fine mesh to the medium mesh. The energy balance equation is,

$$\begin{aligned}\Delta T_{i,j,k}^{n+1} = Fo[&T_{i+1,j,k}^{n} + T_{i-1,j,k}^{n} + T_{i,j+1,k}^{n} + T_{i,j-1,k}^{n} + T_{i,j,k-1}^{n} \\ &+ 1/2\, T_{coarse}^{n} - (5 + 1/2)T_{i,j,k}^{n}]\end{aligned} \tag{2.12}$$

As can be seen in equation 2.12, the factor of 1/2 is present for the same reason as in the fine to medium interface. It should be pointed out that the Fourier number for each mesh is different due to the difference in nodal spacing. The Fourier number in an equation is always calculated for the spacing of the nodal point being evaluated.

d. The Coarse Zone to the Medium Zone

The coarse to medium interface is handled in the same manner as the medium to fine interface. There are 30 adjacent medium nodes to one coarse interface node and these nodal temperature are averaged together:

$$\begin{aligned}\Delta T_{i,j}^{n+1} = Fo[&T_{i+1,jn} + T_{i-1,j}^{n} + T_{i,j+1}^{n} + 3/2\, T_{medium}^{n} + 2BiT_{inf} \\ &- (3 + 3/2 + 2Bi)T_{i,j}^{n}]\end{aligned} \tag{2.13}$$

Equation 2.13 is the energy balance equation for this case. Again, the pseudo node is only 2 3 the distance away as a normal coarse node and the factor 3 2 appears. Equation 2.13 shows the relationship at the interface in the '-j' direction.

D. SIMULATING THE ARC

Simulating the arc and the weld pool is one of the most difficult aspects of the model. Many researchers shape the arc to obtain the desired empirical results of weld pool size, shape and penetration. Such techniques, however, are not appropriate for the study of transients for they only hold in the steady-state. It is desired to add heat to the weld with little or no shaping and yet still model known weld pool dynamics. Two different approaches are taken, both having appropriate applications.

In the first approach the arc and the weld pool are simulated by spatially shaping the forcing function. Based on the experiments of Lu [Ref. 4] the energy of the arc can be assumed Gaussian in distribution. This is then added volumetrically to the fine mesh. The arc is hemispherical with a radius of four millimeters. Further shaping of the arc by use of double ellipsoids as discussed by Goldak [Ref. 5] are not used because it interfered with the study of transients. Since this weld pool simulation shapes the weld pool it is only useful for studying transients that do not pass through the weld pool nor effect weld penetration. The weld pools generated are shallow, with little penetration. This is typical of reverse circulation weld pools where flow is from the center of the pool surface to the edge of the pool.

In the second approach, the arc is given a Gaussian energy distribution in the surface plane but the energy is added only to surface nodes. To simulate the penetrating effect of the arc pressure, the thermal conductivity is made directional. Vertical thermal conductivity is an order of magnitude greater than the horizontal. The directional thermal conductivity only applies in the molten metal temperature range. The weld pools generated showed deeper penetration, which is typical of normal circulation, where the flow is from the center to the bottom of the weld pool.

In both cases the program allowed for the arc to move in the X-Y surface plane. The amount of heat added to each node is determined by calculating the Gaussian distribution factor for each node and summing them. This provided a normalizing volume by which to divide the arc input energy for that time increment. The normalized arc power density is then used to calculate the enthalpy change of each node.

E. APPLYING RUNGE-KUTTA

The application of Runge-Kutta to the explicit finite difference problem proved to be a very powerful tool. Many difficult problems have been successfully solved using explicit Runge-Kutta as described in [Ref. 6]. In appendix B it is shown that for the constant coefficient, three dimensional case the limitation imposed by the Fourier number is eliminated altogether. Thus it is possible to take larger time steps and still maintain accuracy. However, due to the large temperature gradients in the vicinity of the heat source, it is found that time steps in excess of the Fourier number limitation had a rapid degradation in the accuracy and stability of the solution.

The general solution technique of Runge-Kutta involves solving the equations four times for each time step. The advantage is that this provides an accuracy of $O(h^4)$, a substantial improvement over the $O(h^1)$ of using forward differencing. Numeric testing showed that about 1/20 the time step would be required to obtain the same accuracy (even with implicit), so that Runge-Kutta resulted in a times savings of at least 80 percent.

F. PROGRAM OVERVIEW

The program sequence is as follows. The first segment either defines the initial conditions or reloads a previous problem. The main processing portion consists of three parts. First the heat of the arc is added to the fine mesh. Second the change in temperature is calculated by applying Runge-Kutta. Third, the mesh is checked to see if it is time to move meshes. If it is, the meshes are moved and the pointer updated. The last two parts are associated with output. One section handles running output during execution and the last portion saves the final temperature distributions and data necessary to restart the problem, if desired.

There are three major subroutines in the program. These are labeled FIN, MED and COR for fine, medium and coarse and are where the finite difference equations are set up. These are called by the Runge-Kutta solver. There are also three functions. One finds the thermal conductivity as a function of temperature and the other two are used for converting to and from enthalpy and temperature.

III. VALIDATION OF THE NUMERIC MODEL

The accuracy of the model in reflecting the actual welding conditions is heavily dependant on the assumptions made in stating the problem, the accuracy of the material properties and the resolution of the numeric technique. In addition, it is desirable to validate the results of the model against other models and analytical solutions were they are available and against experimental results. This chapter will discuss each of these areas in some detail.

A. VALIDATION OF THE MODEL ASSUMPTIONS

The assumptions of the model are discussed in detail in each of the following sections. The decisions leading to these assumptions were based on the work of previous investigators, Goldak [Ref. 5: pp. 587-600] having provided the best summary in his article. In addition, several simple two dimensional models were used to confirm the accuracy and stability of several solution techniques to further refine the approximation. There are two conflicting desires in developing any model, the first is to maximize the resolution of the solution and the second is to minimize the computational effort. To achieve this, it is necessary to insure that greater resolution is applied only were it is warranted.

Perhaps the first validation of the model is to look at the actual results of the model. Figure 5 shows the surface temperature variation for the x-y plane of the fine zone during a typical startup, Figure 6 is a contour plot of the same startup, and Figure 7 is a contour plot of the x-z plane directly under the arc. These all show the expected shapes of the weld pool and thermal contours during a startup sequence. In the surface profiles, the portion of the weld pool not under the arc is seen as a region of near constant temperature due to the phase change occurring. In all of these pictures the arc is at a y position of about 17 mm.

1. Fine mesh spacing of one millimeter provides adequate resolution

This is the recommended spacing of Goldak, though other researchers have successfully used larger spacings. Referring back to Figure 5, it can be seen that this spacing in the fine zone is more than adequate to produce a smooth thermal contour of the temperature. However, in those cases were it is desired to measure a temperature difference from the quasi-steadystate value, such as flaw detection, some model noise was notice that could have been limited by decreasing the mesh spacing.

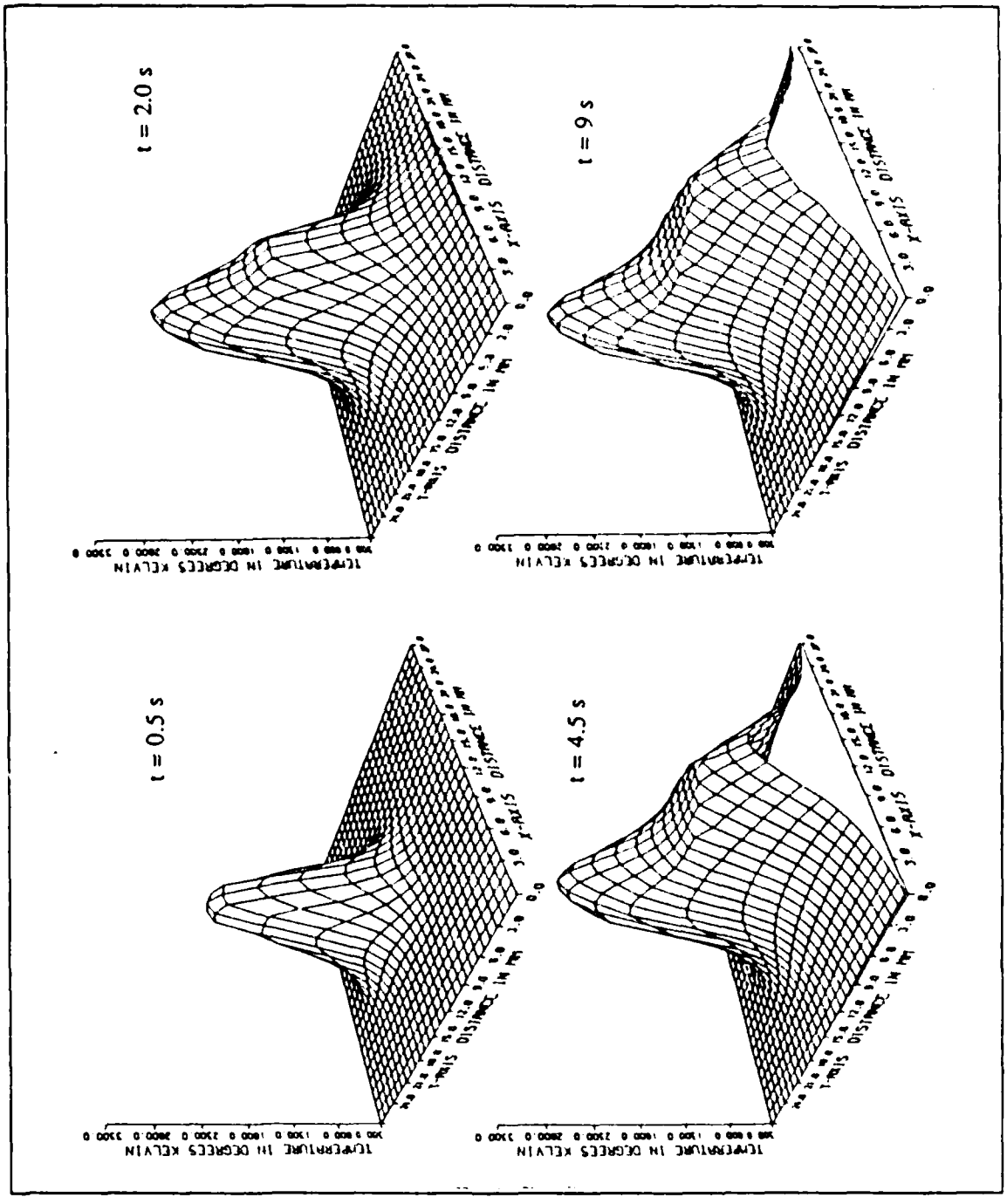


Figure 5. Surface Temperature Variation in the x-y Plane: The torch moves parallel to the fine y axis at x = 14 mm. Q = 2544 watts, v = 4 mm s.

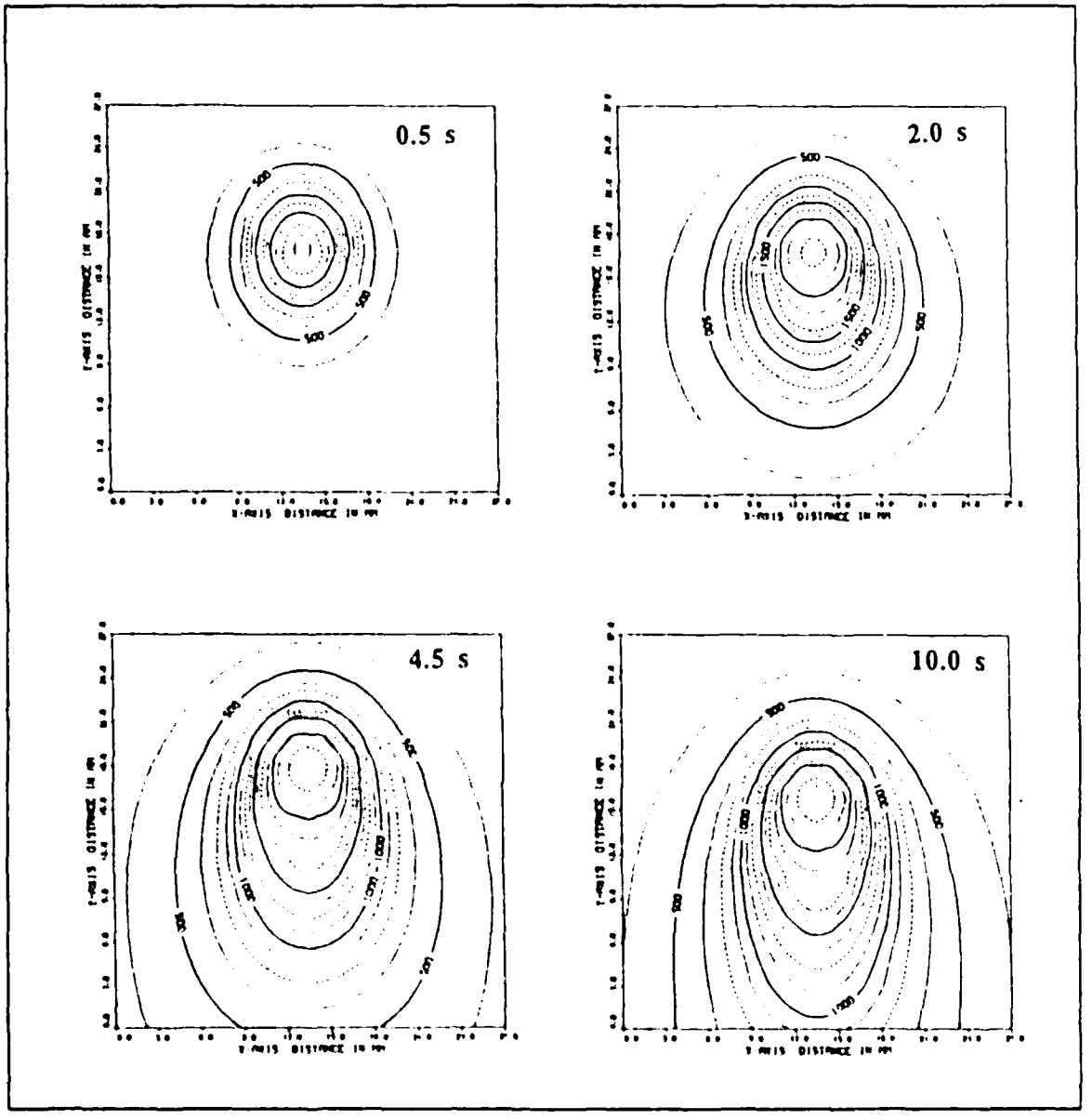


Figure 6. Surface Temperature Contours in the x-y Plane: The torch moves parallel to the y axis at x = 14 mm. The x-y coordinate system is fixed to the fine zone. Q = 2544 watts, v = 4 mm/s.

2. Coarse and Medium meshes decrease the number of nodes

The idea of using different mesh sizes is to minimize the the number of nodes[Ref. 7]. Unlike the Finite Element Method (FEM) it is not a good practice to gradually change the mesh size for Finite Difference. This is because the accuracy of the

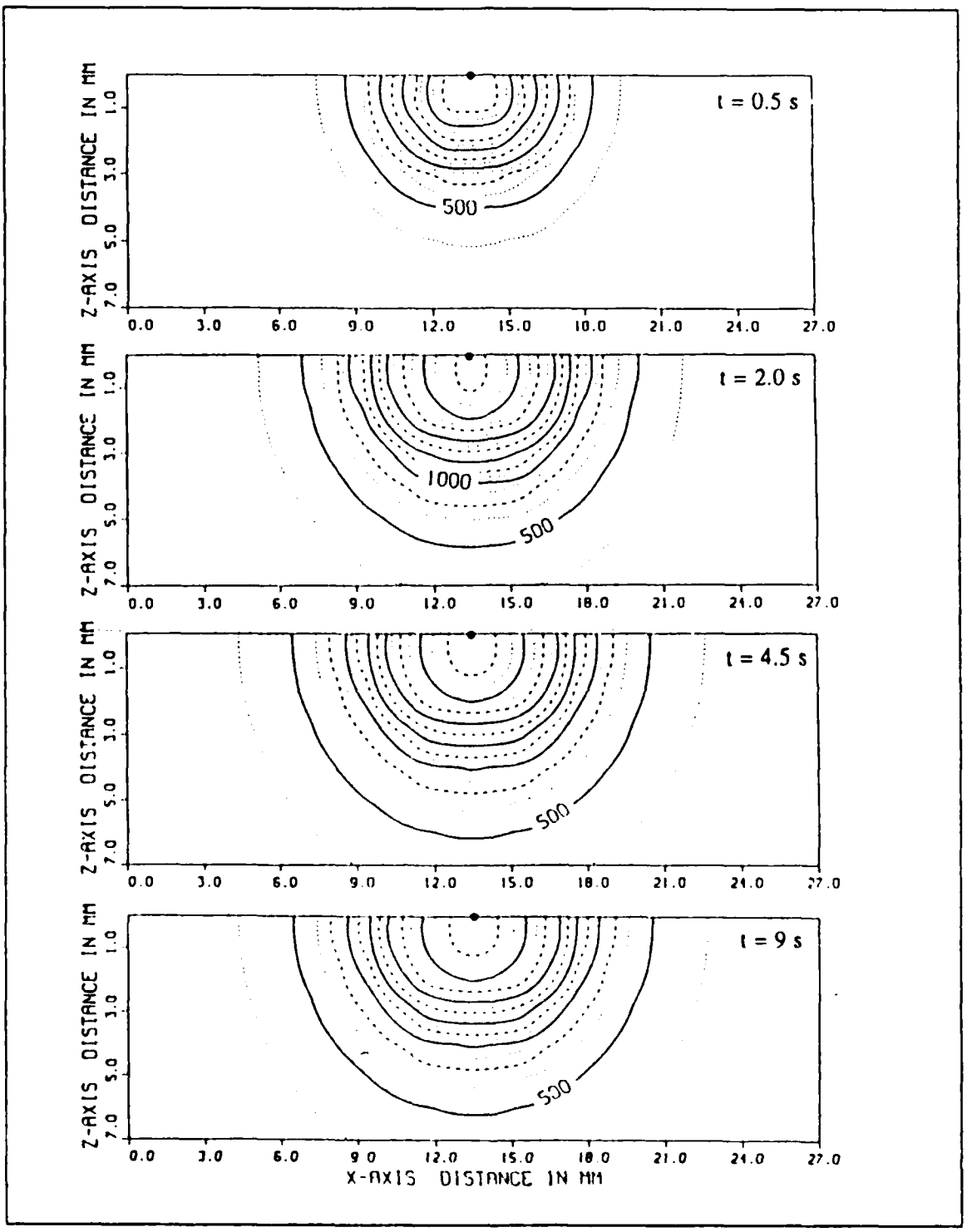


Figure 7. Temperature Contours in the x-z Plane Intersecting the Arc

Finite Difference Method is $O(H^2)$ for uniform mesh sizes versus $O(H)$ for variable mesh sizes. It is necessary to insure that the gradients are low when shifting to a coarser mesh to minimize the error induced by the volumetric averaging. This is checked visually by studying the temperature profiles that the model generates. For instance, in the case of studying cooling rates, it was found that the region behind the arc needed to be increased to allow the cooling rates to be determined more accurately. This particular modification is discussed in further detail in this chapter and caused the program to take approximately twice as long to run.

3. Radiation could be approximated by a linear coefficient

This was a simplification used to improve the speed of the program. It was based on the fact that for a thick plate, conduction dominates the problem compared to radiation and that in this regions, the surface temperature is below the temperature at which significant heat will be lost in radiation. Any surface temperature losses were simulated by use of h, the convection term. This was applied only in the medium and coarse regions.

4. The Free Convection Coefficient may be approximated as a constant

Again, this is a simplification for two reasons. The first is that it speeds up the operation of the program and the second is that the value of h in the welding environment is not accurately known anyway. In addition, this effect is small compared to conduction for the thick steel plate being modeled. Work done by Goldak [Ref. 5: pp. 587-600] confirms the practicality of this assumption.

5. The energy of the arc may be added as a volumetric gaussian source.

This is discussed in greater detail in Chapter 2 where it was mentioned that this has been demonstrated both by other models and experimental work on arc power distributions. In this model, no attempt was made to shape the arc power distribution to obtain a desired matching with an experimental weld pool. Arc shaping was avoided to since the response of the weld pool was one of the desired results and this would be obscured if shaping was used.

6. Material properties are constant far from the arc

This is used in the medium and coarse regions since all of the elevated temperatures are contained within the fine region. Thus, over the lower temperatures that are experienced in the these regions, this is a fair approximation. This has a significant impact on program speed, since it is much faster to calculate the finite difference equation with constant coefficients.

B. INITIAL PROGRAM DEBUGGING

In writing a program from scratch a large number of programming errors occurred, most of which were clerical in nature. These errors were found by letting the program run and observing the results, in most cases the mistakes clearly stood out as violations of the the laws of thermodymanics. Next it was desired to insure that the arc energy was correctly inputted. A simple energy balance was performed to insure that the heat from the arc is conserved. After the model passed these simple tests it was considered free of basic programming errors.

C. COMPARISON TO AN ANALYTICAL SOLUTION

The analytical solution available is called Rosenthal's Solution, and it solves the temperature distribution due to a point source of energy moving along the surface of a semi-infinite solid at uniform velocity. Rosenthal [Ref. 8] describes the derivation of this solution in detail and the result is presented as equation (3.1).

$$T - T_0 = \frac{q}{2\pi kr} e^{-v(r+x)/2\alpha} \tag{3.1}$$

where

$$r = \sqrt{x^2 + y^2 + z^2}$$

and v is the velocity of the arc. The coordinate system is centered at and moves with the arc. This elegant solution is valid for only constant thermal properties. Thus, for the purposes of validation, the model is modified accordingly. All of the thermal properties are made constant (thermal conductivity and the heat capacity). The arc power is reduced to avoid melting and heat is added to a single node instead of using a volumetric source. Also, since Rosenthal's solution does not take into account radiation and convection, these are eliminated from the model.

The results of the validation are shown in Figure 8, which is a comparison of Rosenthal's solution to the model. The current arc position is $y = 17$, and the temperature contour plotted is at $x = z = 0$. This demonstrates that the finite difference model is converging to the analytical solution. The difference seen is the expected one, since any finite difference model will be stiffer than an analytical one. The largest source of error is at the discontinuity at the center of the arc, which is expected due to the point source. If the model is attempting to simulate a point source, it is clear that a finer or different mesh scheme would be required near the arc. However, since it is desired to

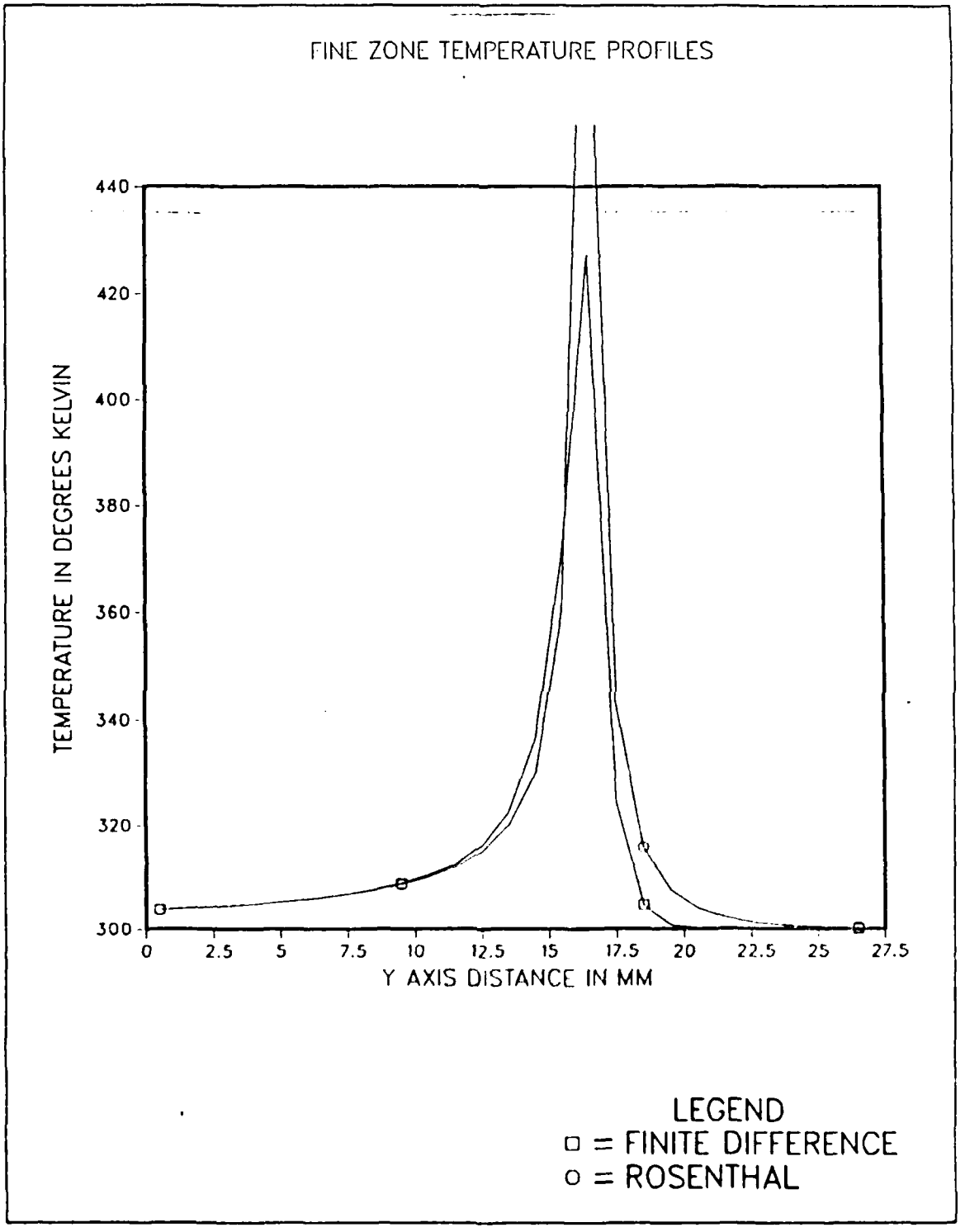


Figure 8. Rosenthal's Solution Compared to the Finite Difference Model

model a volumetric source rather than a point source, the present mesh size is adequate. This is supported by Goldak [Ref. 5: p. 590] who reports that having at least four elements in the arc is adequate and eight is preferable. The spacing of 1 millimeter provides a minimum of eight nodal points under the arc.

D. QUASI-STEADYSTATE THERMAL PROFILES

There are two comparisons that can be easily made to experimental data. This is the measured cooling rates and the weld pool dimensions. Both of the theoretical and experimental work is summarized by Lancaster in his text on welding [Ref. 9]. In general, the analytical model provides results within a factor of 2 of the experimental results. To characterize the weld, a non-dimensional parameter, n, is defined by equation (3.2).

$$n = \frac{qv}{4\pi\alpha^2 C_p(\theta_c - \theta_0)} \tag{3.2}$$

Average values of the weld parameters are:

$$q = 2544\ watts$$
$$v = 0.004\ m/s$$
$$\alpha = 9.0x10^{-6}\ m^2/s$$
$$C_p = 4x10^6\ watts/m^3K$$
$$\theta_c = reference\ temperature$$
$$\theta_0 = 300\ degrees\ K,\ initial\ temperature$$

Using the melting point of steel, 1723 degrees Kelvin, as the reference temperature, it is found that n is 1.8. Referring to figure 3.17 of Lancaster, the weld pool width is eight millimeters, which is exactly what is predicted by the model.

As discussed by Lancaster, there is a great deal of data scatter in trying to determine cooling rates, and in general, the experimental cooling rates are less than the theoretical. This is only in part due to the finite size and time delays associated with thermocouples. To study this, a modified version of the welding program called WELDC was used. As discussed in Appendix C, this program uses extended fine and medium regions to improve the accuracy of cooling rate measurements, since the temperature at which we wish to measure the cooling rate typically occurs 30 millimeters behind the arc. Figure 9shows the cooling rate at the arc centerline for an arc startup and shut down transient. The reference temperature is 810 degrees Kelvin and the initial temperature is 300 degrees Kelvin. In this case, the arc power was only 1950 watts so that the

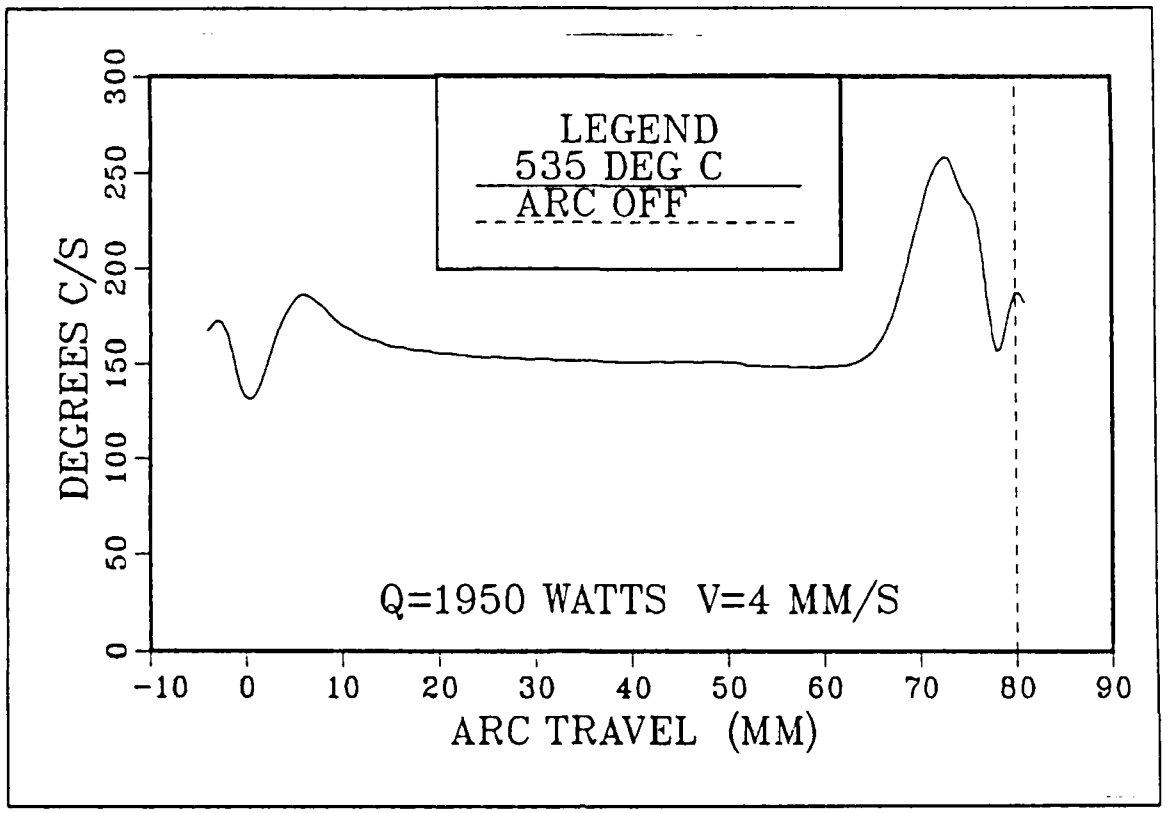


Figure 9. The Cooling Rate Measured at 535 deg C: Shown for an arc startup and shutdown sequence, taken along the arc centerline.

characteristic parameter n was 3.8. Referring to figure 3.19 of Lancaster, the expected value is 140 degrees C per second. This is in excellent agreement with the quasi-steadystate value shown in Figure 9. The difference of ten percent is easily explained due to the error in averaging the material thermal properties in calculating the characteristic parameter n as well as the experimental data scatter.

E. EFFECT OF ARC OSCILLATION

Most automatic welders use an oscillating arc to improve the weld quality. This is because the transverse oscillation insures that the weld pool is spanning both sides of the weld seam, preventing a lack of fusion. A trial run was performed to compare the thermal signature of a steady weld to the case of an oscillating one. For this test the arc was given a sinusoidal oscillation at four hertz and a peak to peak transverse amplitude of one millimeter. The results of this are shown in Figure 10 and as expected the oscillating arc has a wider thermal profile and slightly lower peak temperatures. This again

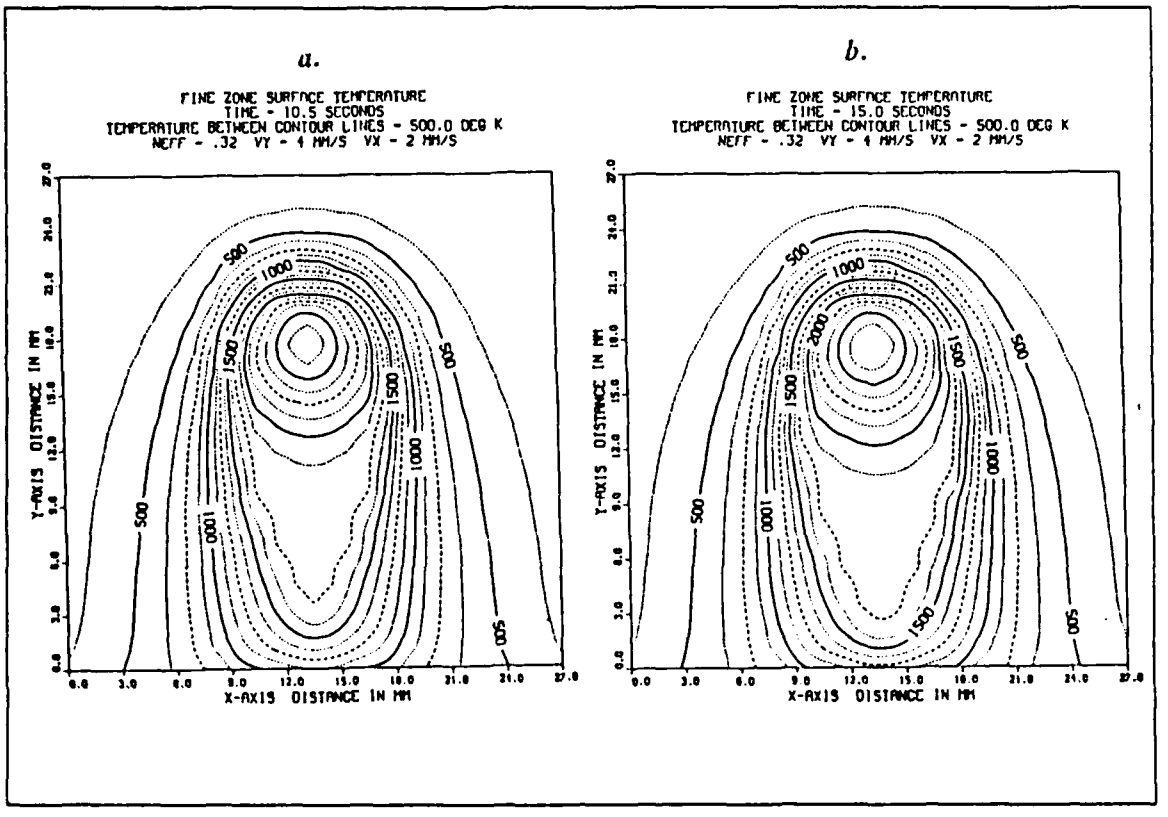


Figure 10. Comparison of a Steady Versus an Oscillating Arc.: a. Steady arc temperature contour. b. Oscillating arc temperature contour.

shows the usefulness of this model, in that it can easily simulate complex arc patterns since it does not invoke symmetry in the geometry.

F. DETERMINING THE MATERIAL PROPERTIES

All of the initial results of the model are based on using the material properties of a low carbon steel provided by Goldak [Ref. 5: p. 591]. However, since the actual test pieces were made of HY-80 steel it was desired to use the properties of this steel also. The data available on the HY series of steels was provided by Morris [Ref. 10]. This data provided thermal conductivity up to 811 degrees K and specific heat up to 673 degrees K. The density of the material is considered to be constant and is about $7890 kg/m^3$ The thermal coefficient of linear expansion up to 1100 degrees F is 7.1×10^{-6} in./in./F. which is less than a one percent change. Above this there is the phase change to face centered cubic where the metal actually contracts, hence considering the density constant is a good approximation [Ref. 11: p. 13].

The thermal conductivity of metals in general does not have a theoretical basis at low temperatures but experimental work has shown that above 1200 degrees K for iron based alloys, all alloys have the same thermal conductivity. In Figure 11 is the plot of thermal conductivity for pure iron, a stainless steel and for HY-80 steel. Since HY-80 has two percent chromium it is expected to lie between pure iron and the stainless steel since alloy concentration is a major parameter at lower temperatures affecting the thermal conductivity [Ref. 12]. Thus with a high degree of confidence the thermal conductivity of HY-80 steel can be extrapolated to the melting point.

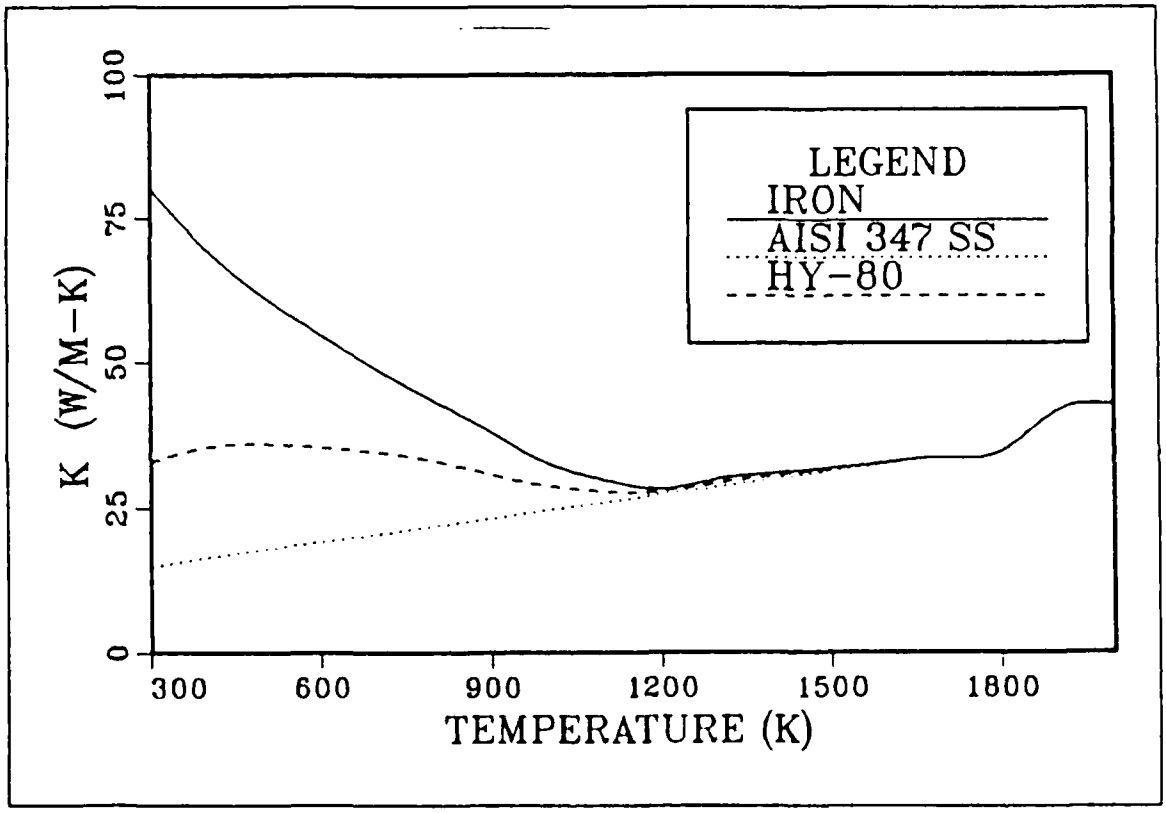


Figure 11. Thermal Conductivity versus Temperature of Iron Based Alloys

The theoretical limit of heat capacity is well known as 3R (6 *cal/mole °K*) and is not significantly affected by alloying, except where heats of transformations are concerned. No good values exist for high temperature steels in general. This problem is further complicated by the fact that the heat of transformation is not just a function of the temperature but also on the material thermal history and the present rate of temperature change. The difference in energy stored in a martensitic steel versus a pearlitic steel is not negligible. Due to these factors, a precise determination of the heat capacity of HY-80 steel would be of limited value. The average value of Cp for Goldak from 0 to

650 degree C is 536 J/kg C while a typical value from Morris is 522 J/kg C. Thus, to continue to use the values of Goldak for a low carbon steel is appropriate, since the data provided by Morris is only to 400 degrees C.

G. CONCLUSIONS

The use of Runge-Kutta and the explicit form of the finite difference equations is a powerful tool for three dimensional modeling of welding. For the unique case where the temperature gradient is controlling the accuracy this is the fastest way to solve this problem. In addition, it greatly allows expanding the number of nodes in the problem since the computational cost is linear. Even techniques such as Gauss-Seidel for solving a large sparse matrix will take longer to converge the sparser the matrix is and then only provide an approximate solution, with no advantage gained since the gradient still limits the time step.

The weld model as implemented takes advantage of the lower cost of using constant properties were the error in doing so is small. In the regions of elevated temperatures, variable thermal properties are employed. The data available for the temperature dependence of the properties of steel is adequate for reasonable accuracy. The use of empirical weld pool results is used in lieu of simultaneously solving the weld pool dynamics. The later problem is compounded by the fact the altering the weld pool dynamics is the major reason for welding rod doping. Due to this, no advantage is to be gained by attempting to model the weld pool dynamics, since the final weld pool shape is controlled by the metallurgist. Thus, for the weld parameters at which it was tested, the assumptions are valid and the resulting temperature profiles simulate the experimental results.

Because of the models inherent simple structure it is straight foreword to alter. There is no mesh generator, each element can be directly altered by the user to allow simulating weld flaws, torch to weld seam misalignment and other items of interest. By not using symmetry in developing the model, the program is able to study these asymmetrical problems. The use of explicit finite difference allowed the nodes to be described by three dimensional arrays, so that there is no nodal numbering problem which is normally associated with using implicit techniques. This lets the user directly see what is happening at each node in the model, as well as the ability for simply changing its condition. This fact was exploited in the modified versions of the weld program, where the modifications were merely inserted and required little additional programming.

IV. SEAM TRACKING DURING WELDING

A. BACKGROUND

Infrared optical systems have been used successfully for weld seam tracking by detecting the surface thermal profiles. This is discussed in detail by Begin in [Ref. 13: pp. 518-522] and Khan in [Ref. 14: pp. 799-805]. In both cases this was experimental work with no computer modeling. Since experimental results were available for comparison it was desired to model the occurrence of the welding arc misalignment.

B. MODIFICATIONS TO THE WELDING MODEL

To allow the occurrence of arc misalignment three basic steps were taken. First, a weld seam had to be simulated in the material. Second, the arc must reach steady state and then be programmed to track off to one side of the seam. And third, a run was conducted with a track off, with no seam present, so that the change in direction could be removed as a factor in the resulting thermal profile.

The weld seam was simulated as a butt joint of two plates. The interface was a region of zero thermal conductivity. This was created only in the fine zone region and was done by placing a logic statement to check to see if the region in the fine zone should have zero thermal conductivity or not. This check also tracked the thermal history of the nodes adjacent to the butt joint. If the temperature of either node had exceeded the melting temperature of the metal, the thermal conductivity was returned to normal, simulating that the material had fused.

The modified programs for the model are appended with MA for misalignment. The FINMA subroutine is altered to include a logical variable called MELT which is used to determine which portions of the seam have melted. Since the seam is simulated between nodes with x locations of 13 and 14, only the temperature history of these nodes are monitored. In Appendix A a detailed description of the model is given, the modifications are to only two parts of the FINMA subroutine. This is for all internal nodes nd for top surface nodes. In Appendix C is the listing of the modified code. Basically the fine node enthalpy changes are calculated normally for all nodes. The temperatures of the nodes adjacent to the seam are checked to see if either has melted, if they have, the logical variable MELT is set equal to true for that seam location. Then for those nodes adjacent to the parts of the seam which is not melted, the change in enthalpy is recalculated using a zero thermal conductivity across the seam. This approach is used

since it required the least modification of the code. The variable MELT is passed to the WELDMA main program to allow it to be saved for program restart.

To simulate the misalignment the arc is started centered at a position of 13.5, directly over the seam. The arc is started and tracks the seam until quasi-steadystate is reached . The arc is now shifted off to the right at .5 mm/s with a base speed of 4 mm/s. The surface thermal profile experiences a dramatic shift very soon after the arc is no longer centered on the seam. This is consistent with the results of experiments in seam tracking.

C. MODEL SIMULATION RESULTS

In Figure 12 the case of a weld misalignment is shown with a seam present. Here a very dramatic change is seen, especially relative to the axis perpendicular to the weld axis. This change in the surface temperature closely matches the observed experimental result. It is important to note that the misalignment is first detected before the arc. In fact, due to the size of the weld pool, the weld quality may still be satisfactory when the misalignment is detected. In this case, there is time to effect corrective action.

The model also shows that the surface temperature profile is dominated by the shallow region near the surface. In these simulations, the weld seam is only 6.5 millimeters deep, the rest of the 20.5 millimeters is solid. This implies that even for multipass welds, a shallow seam is adequate for seam tracking. It also indicates that tracking the seam successfully says little about weld penetration. This later concern has been studied, and it was found that weld penetration could be correlated to weld pool diameter and the peak weld pool temperature.

D. CONCLUSIONS

The major effect noted is a rapid decrease in temperature on the side of the arc that the seam is on. This is due to the poor conduction across the butt joint if melting does not occur. This effect is important since very primitive thermal sensors are capable of detecting this type of change. Extensive testing with flaws, as discussed in Chapter 5, has shown that flaws in general always cause a rise in all temperatures in the plate due to the overall reduction in thermal conduction. Since the only flaws which caused a decrease in temperature were equivalent to localized misalignments, it can be accepted that a rapid decrease in temperature on one side of the arc is caused by the arc moving off of the seam. In addition, subsurface flaws are rarely detected prior to the arrival of the arc. By monitoring the temperature ahead of the arc, corrective action can then be correctly taken to redirect the arc toward the colder side. This is in agreement with the

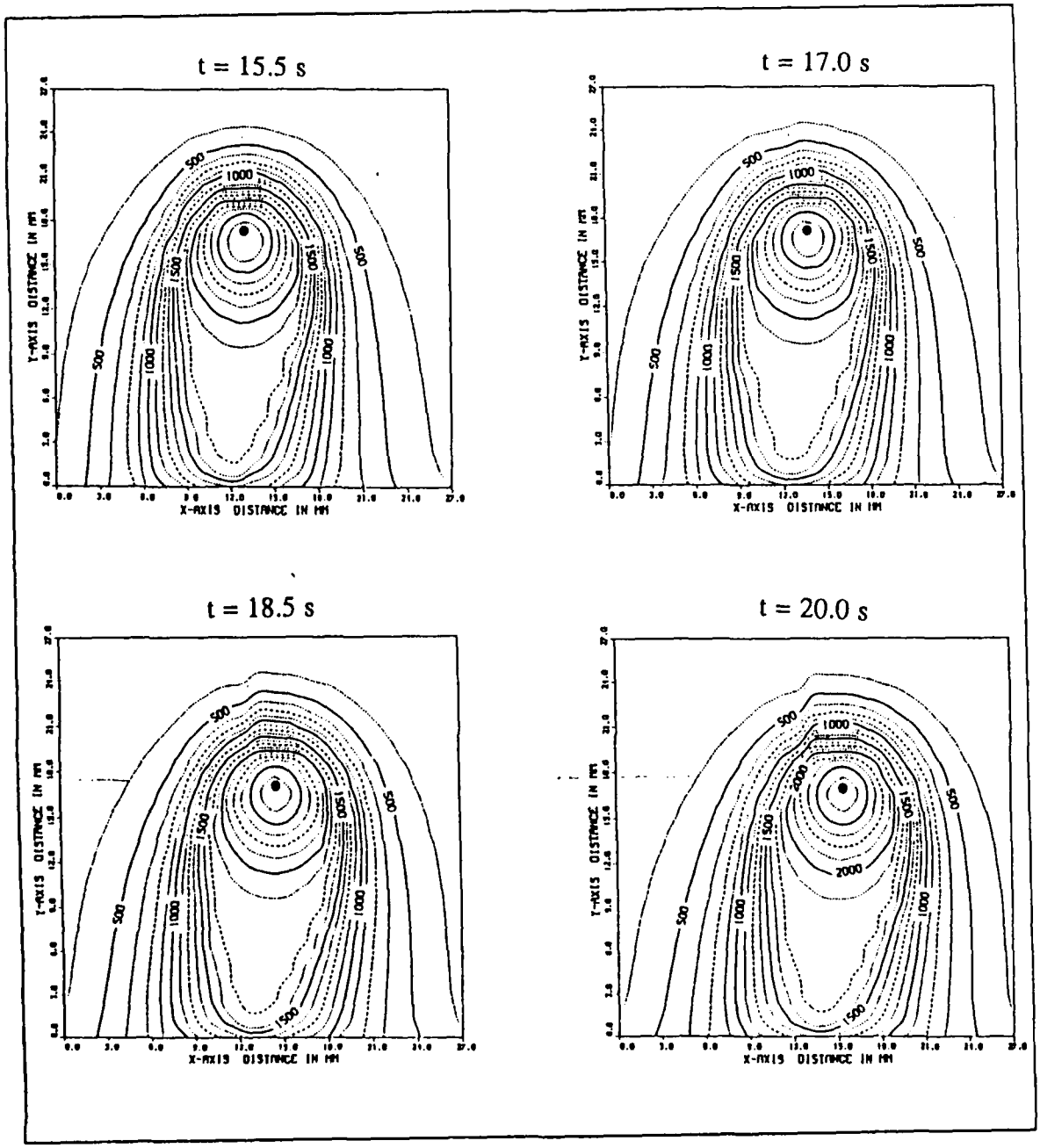


Figure 12. Torch Misalignment Modification to Surface Temperature Contours: After moving along the y axis for 15 s with v_y = 4 mm/s, the torch veers off to the right with an additional velocity component of v_x = 0.5 mm/s. Q = 2544 watts.

experimental success in seam tracking. In addition, due to the magnitude of the transient that occurs, industrially suitable low resolution sensors, such as infrared diodes,

could be used in lieu of an expensive and delicate camera system. This would provide a low cost and less intrusive approach. This concept is discussed further in Chapter 7.

V. FLAW DETECTION DURING WELDING

A. BACKGROUND

The prevention and detection of weld defects is the major source of cost associated with large scale arc welding. Due to the low metal deposition rates that are allowed when GMA welding the new HSLA series steels, weld quality becomes even more critical. To support the expert welding system, data must be acquired to interpret the meaning of the various thermal signatures that any infrared imaging system would produce of the arc welding process. It is desired to know the detectability of a given subsurface flaw for a given set of welding conditions to determine if the flaw is detectable, and if so, how does it change the quasi-steadystate thermal profile?

Little experimental work has been done in flaw detection using a infrared camera system. Khan in [Ref. 14: p. 801] performed experiments in detecting surface defects and reported that a small Al_2O_3 particle on the surface was detectable by a distortion of the temperature isotherms of the weld pool. These perturbations would start as early as 1.5 inches before the center of the arc passed over the flaw. Since the low thermal conductivity impurity was on the surface, it always shows as a cold spot in the surface temperature field.

No analytical work has been performed in flaw detection. Current weld models rely heavily on symmetry, and flaws are generally asymmetrical. Since the current model does not use symmetry, it is ideally suited for studying this type of problem. A large number of flaws was simulated using a modified code, which is designated with the suffix, LF, for Lack of Fusion. These data runs were then reduced to develop a single correlation which could be used to predict the detectability of subsurface flaws.

B. MODIFICATION OF THE WELDING MODEL

The flaw is simulated by a group of nodes which had zero thermal conductivity. The location of these nodes was restricted to the internal and top surface nodes of the fine zone for ease of programming. The basic source code is explained in detail in Appendix A and the modified source code is listed in Appendix C. The flaw is rectangular in shape and its size and shape are specified by inputting the X and Z coordinates of the corners of the flaw plus its length in the Y direction. These five input variables are labeled I1, I2, K1, K2, and JL. I1 and I2 are the limits of the flaw in the X direction, K1 and K2 are the limits in the Z direction and JL is the length of the flaw in the Y

direction. Note, since the arc moves in the Y direction, there are no fixed Y limits. The relative position of the arc to the flaw is determined by a variable called LOF (Lack Of Fusion). LOF is calculated by equation 5.1,

$$LOF = 65 - INT(VEL \times TIME) \tag{5.1}$$

which is for a velocity of 4 mm/s, and a time of 10 seconds. Thus at time 10, the flaw is at position 25 on the fine zone plate. For different times to quasi-steadystate and different weld speeds, equation 5.1 should be modified in the source code accordingly.

The subroutine FINLF subroutine is modified as little as possible for ease of programming. The thermal conductivity at each node is calculated for the fine zone and saved in a new array called CK. Then the thermal conductivity for those nodes presently at the flaw location are zeroed, using the variable LOF as the position reference. The location of the flaw is limited to nodes with i = 3 to 25, j = 3 to 25 and k = 1 to 6. This is because only the surface and center nodes were modified to allow flaws. Thus, for just the internal and top surface nodes, a new form of the difference equation is used, using a function called HK vice GK defined by equation (A.4).

$$HK(i+1,j,k,x,y,z) = \frac{2CK(i+1,j,k)(T_{i+1,j,k} - T_{x,y,z})}{CK(i+1,j,k) + CK(x,y,z)} \tag{5.2}$$

Equation (5.2) defines HK, the major difference is that it uses the precalculated thermal conductivities vice recalculating them as GK does.

The main program, WELDLF, is modified to allow starting the program with the inputs as the size and location of the flaw. This program was usually run for only 5 second simulations, with several being done in a night. A typical EXEC is shown in appendix C for performing this multiple operation using VMSCHED. WELDLF first produces FILE VERIF and echoes the input and base line data for future reference. A second new file is created called FILE COMP, for a comparison file. Every quarter second of simulation an arc profile is saved. The profile is taken at the node which is three millimeters behind the arc, since experimentation showed that this would be the good place for monitoring the effects of flaws. This data was then processed further in a number of ways to determine the effect of the flaw.

C. COMPUTER MODEL RESULTS

In Figure 13 are contour plots showing the effect of a typical surface flaw. The torch has been traveling for 10 seconds when the flaw is suddenly introduced 5 mm

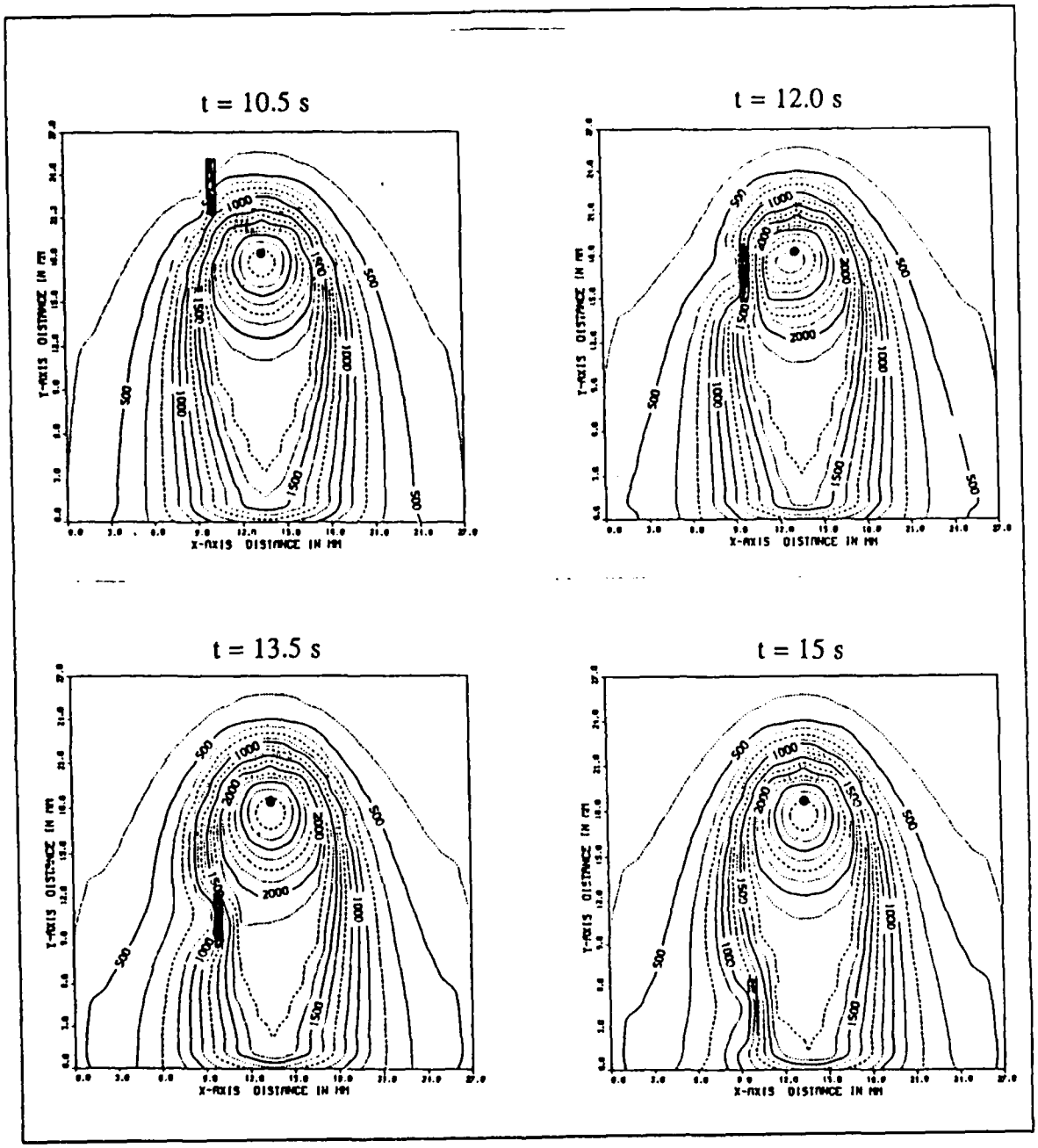


Figure 13. Surface Flaw Modification to the Surface Temperature Contour: A thermally non-conducting inclusion of dimensions $\Delta x = 1$, $\Delta y = 4$, $\Delta z = 4$ and located at $x = 11$ mm, is passed by the arc. At $t = 10$ s the rear surface of the inclusion is 5 mm ahead of the arc center. $Q = 2544$ watts, $v = 4$ mm/s.

ahead of the arc center. The large drop in the local temperatures near the flaw is clearly seen from the distortion in the contours as the arc moves by it. The deviation in temperatures is observable ahead of the flaw and could be used for real time process control. Do to an error in the program, the results that follow used a thermal conductivity only half of that desired for the calculations in the fine zone. Sample runs with the correct thermal conductivity shows that the following results remained qualitatively correct.

The surface temperature changes from the established pattern, resulting due to a relatively deeply embedded non-fusion zone are seen in Figure 14. The temperature change from the quasi-steady levels are seen 3 mm behind the center of the heat source parallel to the x axis for four different flaw sizes. All flaws begin at $x = 11$ mm and $z = 4$ mm and have $\Delta x = 3$ mm and $\Delta z = 1$ mm. At $t = 10$ seconds the front surface of each inclusion is 9 mm ahead of the arc center. The flaw lengths in millimeters are; (a) $\Delta y = 4$, (b) $\Delta y = 3$, (c) $\Delta y = 3$ and (d) $\Delta y = 1$. The arc power is 2544 watts and the torch velocity is 4 millimeters per second. This simulates the response of a linear array of non-contact sensors attached to the welding torch, sensing surface temperatures behind the arc.

Since these flaws are well below the surface, they are detectable by changes in the surface thermal profile only after the arc center passes over the flaw. They are then detected as a local hot spot, apparently due to the lack of conduction of heat through the flaw, raising the nearby interior and surface temperatures. It is clear from comparing Figure 13 and Figure 14 that unlike surface flaws, interior material defects such as slag inclusions or a simple lack of fusion are not easily detected prior to the arc passing over them. Thus while these flaws cannot be prevented by early detection, they may be detected when the arc passes over them. The detection and marking of sites of possible flaws is still a useful aid in improving the weld quality.

The temperature deviations on the surface due to near surface, embedded flaws are seen in Figure 15. All four flaws are located 3 mm to the left of the arc path and begin 2 mm below the surface. Each has $\Delta x = 1$ mm and $\Delta z = 2$ mm. The flaw size in the y direction, Δy, decreases from 4 mm to 1 mm in 1 mm increments as in Figure 14. For all four flaw sizes, there is now a local hot spot between the area and the flaw and a larger local cold spot on the opposite side of the flaw. Temperatures between the area and flaw rise temporarily due to the local accumulation of thermal energy. On the opposite side of the flaw the temperatures drop, since the flaw provides a constriction resistance to heat flow. The power and velocity for this example is the same as before.

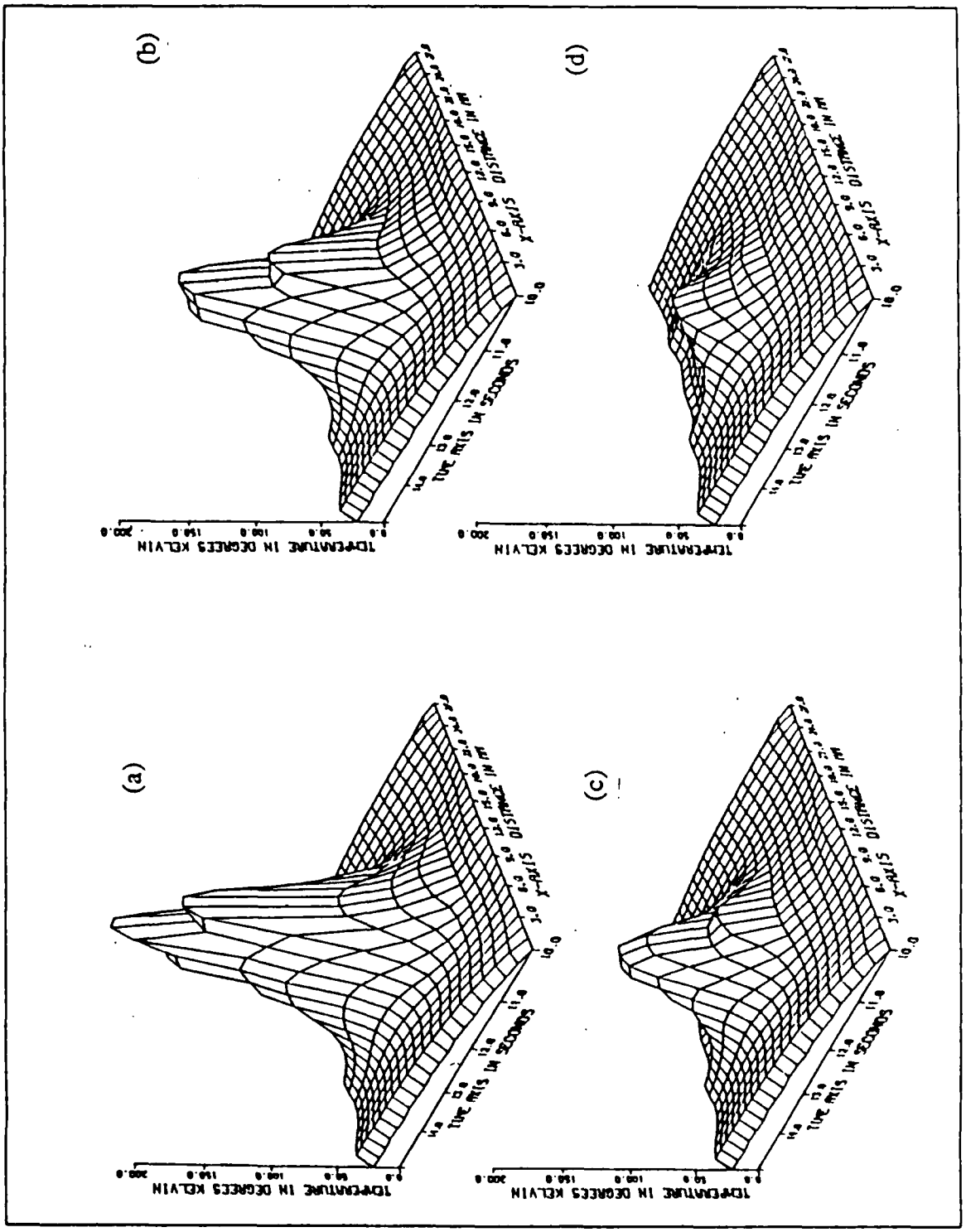


Figure 14. Deep Flaw Modification to the Surface Temperature Profile

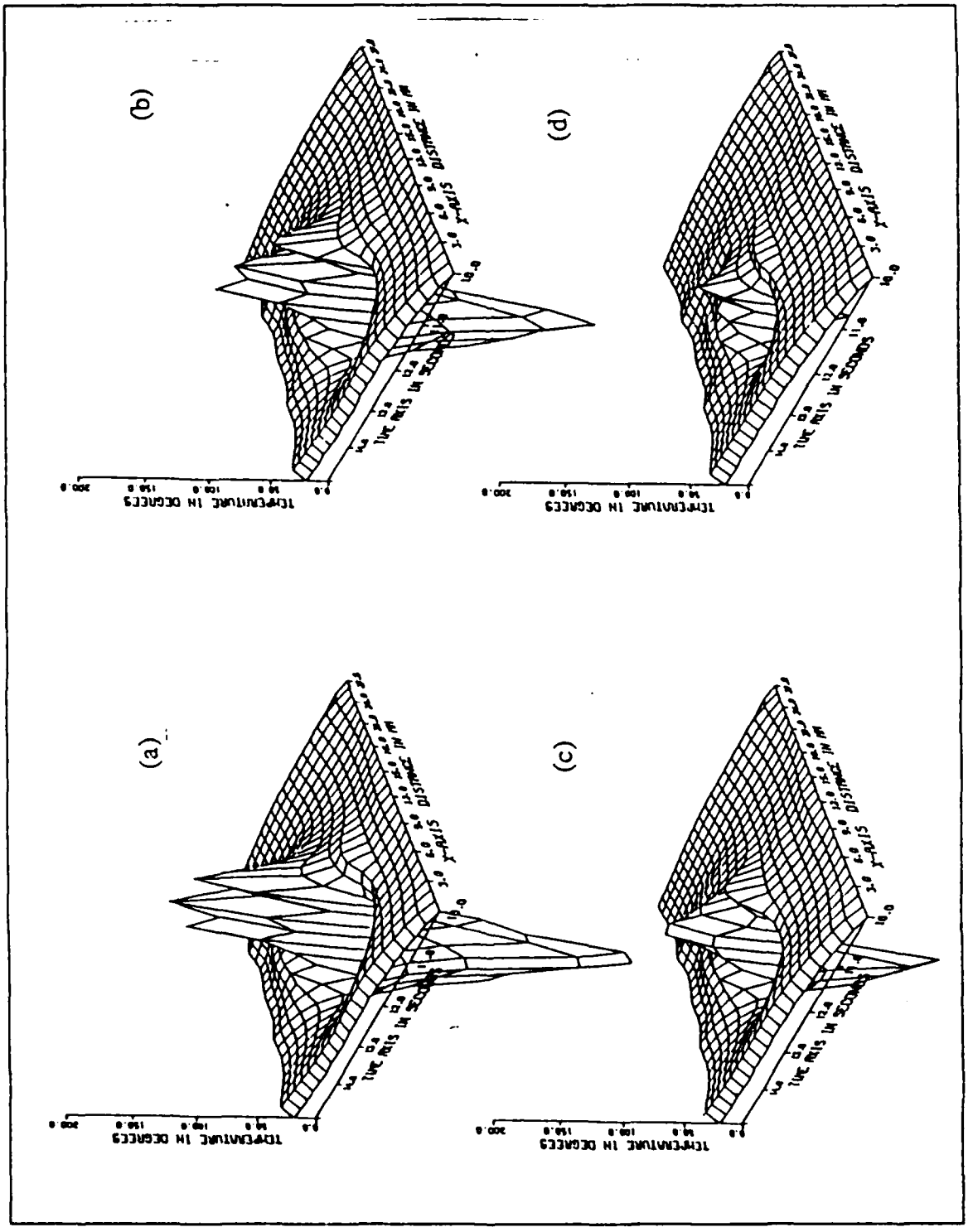


Figure 15. Near Surface Flaw Effect on the Surface Temperature Profile

D. CORRELATION OF FLAW GEOMETRY TO SURFACE TEMPERATURE CHANGES

For a given material and weld parameters it should be possible to establish a correlation between the flaw size and location to its effect on the surface temperature. For plain carbon steel a series of different flaws and geometries were run to develop this correlation. The variables that could effect the surface temperature profile are listed in Table 1. The material properties, arc power and velocity were held constant for each series of trials. The distance behind the arc was selected as a compromise on time response and observed change in temperatures. For the following result, a scan position of three millimeters behind the arc was used. The other five variables were varied one at a time. The resulting correlation is shown in equation (5.3) and is based on a total of 32 different combinations of flaws and geometries.

$$\left| \frac{T - T_{qss}}{t_{qss}} \right|_{max} = \frac{J A_{xy}}{R Z^2} \tag{5.3}$$

where

$A_{xy} = \Delta X \Delta Y \; mm^2$

$R = \sqrt{X^2 + Z^2} \; mm$

T_{qss} = The quasi-steadystate temperature at an x-y scan location on the surface in degrees C, referenced to the initial temperature.

T = The measured temperature at an x-y scan location on the surface in degrees C.

$J = 23. \; mm$, The correlation constant for Q = 2544 watts, low carbon steel, v = 4 mm/s, Y_s = 3 mm.

The left-hand term of equation (5.3) was selected since it is easy to measure. The temperature profile three millimeters behind the arc is periodically scanned. The maximum relative change at any X location from the quasi-steadystate value of the temperature was found to be an excellent measure of existence of a subsurface flaw. The absolute value is for the case of shallow flaws which cause a larger local cold spot. In this case, the maximum temperature change that occurs is negative. This correlation has three major components:

- The detectability of a given flaw drops off as the cube of its depth in the material.

Table 1. SURFACE TEMPERATURE VARIABLES

Variable	Definition
Q	The heat input rate
ρ	The material density
k	The thermal conductivity
C_p	The thermal heat capacity
v	The arc velocity
X	The lateral distance from the arc path
Y_t	The distance behind the arc the temperature is scanned
Z	The depth of the flaw (measured from flaw top)
ΔX	The width of the flaw
ΔY	The length of the flaw
ΔZ	The height of the flaw

- The detectability of the flaw drops off inversely to the radial distance of the flaw from the arc. This is very similar to Rosenthal's analytical solution for a point source.
- The detectability is directly related to the horizontal area (XY) of the flaw.

The surprising result was that the depth of the flaw, ΔZ, had little effect on the surface temperature profile. This is due to the dominance of the Z cubed term. As the flaw extends deeper into the plate, the effect of the extension drops off as the cube. Thus, this term can be neglected with little loss of accuracy. The above correlation is accurate within about 20 percent for flaws ranging from about 1 cubic millimeter to 50 cubic millimeters. It is theorized that this constant is accurate for similar welding conditions that have identical α and heat input rate, q/v. The difficulty in extending the model is that the thermal properties are very nonlinear exactly at the point were the best temperature changes are occurring. The correlation would be much more accurate if the temperatures were monitored far from the arc. However, the most sensitive region to flaws, are those areas which have the highest thermal gradients. This is because the thermal gradient acts as an amplifier of the perturbation caused by the flaw.

E. EXPERIMENTAL VERIFICATION

A simple experiment was run to attempt to verify a the predictions about the detectability of flaws. A one inch plate of HY-80 steel 7 by 12 inches was used, similar to

the welding model. To simulate a flaw of four square millimeters, 3 16 inch holes were drilled into the plate from the back side to just below the surface. The remaining depths tested were 0.5 and 1.5 millimeters. It was felt that these holes would be an excellent simulation, since the model predicted that the sensitivity would be dominated by the depth below the surface, and not the extent of the flaw. A GTA weld of about 2500 watts was passed by the flaw at about three millimeters per second. An infrared camera with five degree Celsius sensitivity was used to monitor the arc thermal profile as it passed by the holes.

As the model predicted, the results for this case were hard to detect. Some minor deviation in the thermal profile was noted when the arc passed by the flaw. The shallowest flaw had the largest effect, being only a rise in temperature. The predicted drop in temperature was not observed, probably due to the flaw passing into the weld pool. The camera only produced thermal contours and like those produce by the model, these showed little effect from a local subsurface flaw. The temperature changes predicted by the model were on the order of 10 to 50 degrees. To effectively detect this, the model simulated a scanner at a fixed distance behind the arc. The use of an actual device performing this, instead of an infrared camera, will be necessary to reliably detect subsurface flaws.

F. CONCLUSIONS

The model can be used to generate correlations between flaws and the surface temperature profile for any given material and welding parameters. For this particular case, a one cubic millimeter flaw located 4 mm down in the plate and 4 mm off of the arc center line would cause only a 0.3 percent change in the any portion of the temperature profile as the arc passes by the flaw. A flaw this small and at this location would be lost in the noise of a real welding process. However, larger flaws would still be detected.

The modeling of flaws during welding has provided some additional insight into the detectability of flaws. In general, the surface temperature profile will only detect flaws that are associated with the actual weld, such as slag inclusions, in and around the heat affected zone. This is due to its inability to detect beyond several millimeters into the work piece. Such detection and marking of flaws would be valuable for real time monitoring of weld quality and as an aid to post weld inspection. However, due to its limited ability to detect flaws, it is unlikely that it could be used in lieu of current inspection techniques.

VI. THE MEASUREMENT AND CONTROL OF COOLING RATES

Weld quality is dependant upon the cooling rate of the weld metal. The major three parameters of concern are; the yield strength, the nill ductility toughness and hydrogen embrittlement (delayed cracking). The yield strength increases with higher cooling rates. The nill ductility toughness is material dependant. For example, in HY-80 higher cooling rates increase the toughness, while for HY-130, higher cooling rates reduce the toughness. The higher the cooling rates, the more likely hydrogen is to become trapped in the weld metal, increasing brittleness, which for HY-80 can appear as delayed cracking. The new high strength, low alloy steels require even tighter control of their cooling rates than the HY series to have satisfactory weld quality.[Refs. 11: pp. 18-21, 15]

Mr. Morris at David Taylor Research Laboratory requested that the computer model be used to study cooling rates. He was particularly interested in controlling cooling rates during the period of arc start up and shut down. During start up and shut down the cooling rates are far from the steady state values and the material properties are usually unacceptable. Industrial practice is often to have run-on and run-off pieces of material adjacent to the desired weld piece which will be removed after the weld. For automatic welding, this is not often possible or too costly, in which case the start and stop areas are ground out and rewelded by hand. When welding by hand the

> arc starting is also very important. The arc and puddle do not have the full protection of the electrode coating at the instant of starting and porosity can result. A proven and recommended starting procedure is to strike the arc an inch or so ahead and then rapidly back-step to the desired starting spot. In this way, the small amount of metal deposited during the start will be remelted as welding progresses and cleansed of any gas or impurities.[Ref. 11 p.1]

A. THE MODIFICATIONS TO THE WELD MODEL

The original weld program did not directly measure cooling rates. In addition, the temperatures that cooling rates are measured are typically 30 millimeters behind the arc. This required that the welding model be modified. The first modification was to insert a subroutine which would measure and report the cooling rates. The second modification was to increase the size of the fine and medium zones so that most of the measurement would occur in the fine zone, which has the highest accuracy. This modification approximately doubled the number of nodes in the model and hence doubled the

computational effort. The modified version is called WELDC and is included in Appendix C with a more detailed explanation of the modification.

B. CORRELATING COOLING RATES TO MATERIAL PROPERTIES

The first question is how to measure the cooling rate. There are three commonly used standards, all three of which were measured:

- The first is the instantaneous cooling rate at 535 degrees Celsius. This is measured by taking the average over the range of 550 to 520 degrees Celsius. It is considered the baseline cooling rate for determining the properties of the HY series steels.
- The second is the cooling rate at 650 degrees Celsius. This is measured by taking the average over the range of 800 to 500 degrees Celsius. It is correlated to the effect of hydrogen cracking.
- The third is the cooling rate at 325 degrees Celsius. This is measured by taking the average over the range of 400 to 250 degrees Celsius. It is correlated to the material strength.

The limits for these cooling rates are based on measuring them along the arc centerline. Obviously, as one moves away from the center line the cooling rates will be less. It should be noted that even though the cooling rate at 535 degrees Celsius is centered at a lower temperature than the one at 650 degrees Celsius, it in general will have the higher cooling rate because the band it is the average of is so much smaller.

C. THE COOLING RATES DURING STARTUP AND SHUTDOWN

To study the cooling rates an arc power of 1950 watts and a velocity of 4 mm s were selected after some experimentation. This is a fairly slow speed and low power arc, and hence has higher than normally desired cooling rates. Typical production weld cooling rates are about 25 to 20 degrees Celsius per second at 535 degrees Celsius. For the low power used, the cooling rates measured were typically 150 degrees Celsius per second. The reason for using the lower power is so the metal will have cooled to the desired range before leaving the medium zone of the model. Even at this lowered power it was necessary to double the size of the medium and fine zones to obtain this. In addition, the higher the power and the slower the cooling rate, the longer the model simulation is required to run to obtain a complete set of results. For the runs presented below, each was a simulation of 30 seconds of welding and cooling. Each run of 30 seconds took approximately six hours of CPU time to complete. The results of this numerical testing are intended to see how cooling rates during transients in general may be controlled. Due to a programming error, the thermal conductivity in the fine zone was only one half

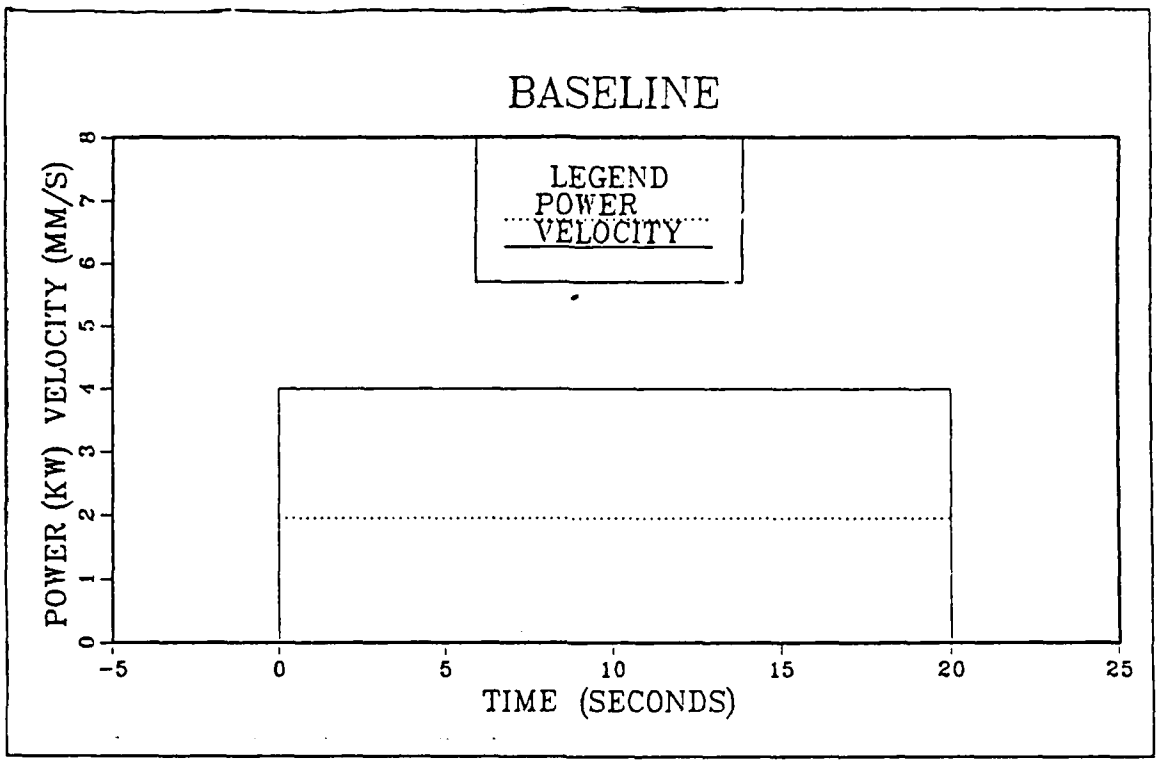


Figure 16. Baseline Velocity and Power Program

of the required. A test run was conducted using the correct thermal conductivity and verified that the following results were qualitatively correct.

The baseline power and velocity program shown in Figure 16 is a simple arc start and stop. The resulting cooling rates are shown in Figure 17. At a position of zero, the arc is turned on and commences to move to the right at 4 mm/s. After a run of 80 mm, the arc is extinguished. At the start of the arc there are very high cooling rates. This is because little energy has yet diffused in the material and hence the thermal gradients are very steep. After the arc has traveled 15 mm, quassi-steadystate is reached and cooling rates are steady. When the arc shuts down, there is a rapid increase in cooling rates again, much for the same reason as at the start. In Figure 17 are also the cooling rates at 535 degrees Celsius for several distances off of the centerline. As expected, as you move away from the arc, the thermal gradient is less steep and the cooling rates go down. These lower cooling rates do not always adversely effect material properties, since the time above the transition temperature is less further out into the heat affected zone. A more graphic display of the pattern of cooling rates is given in Figure 18 where a

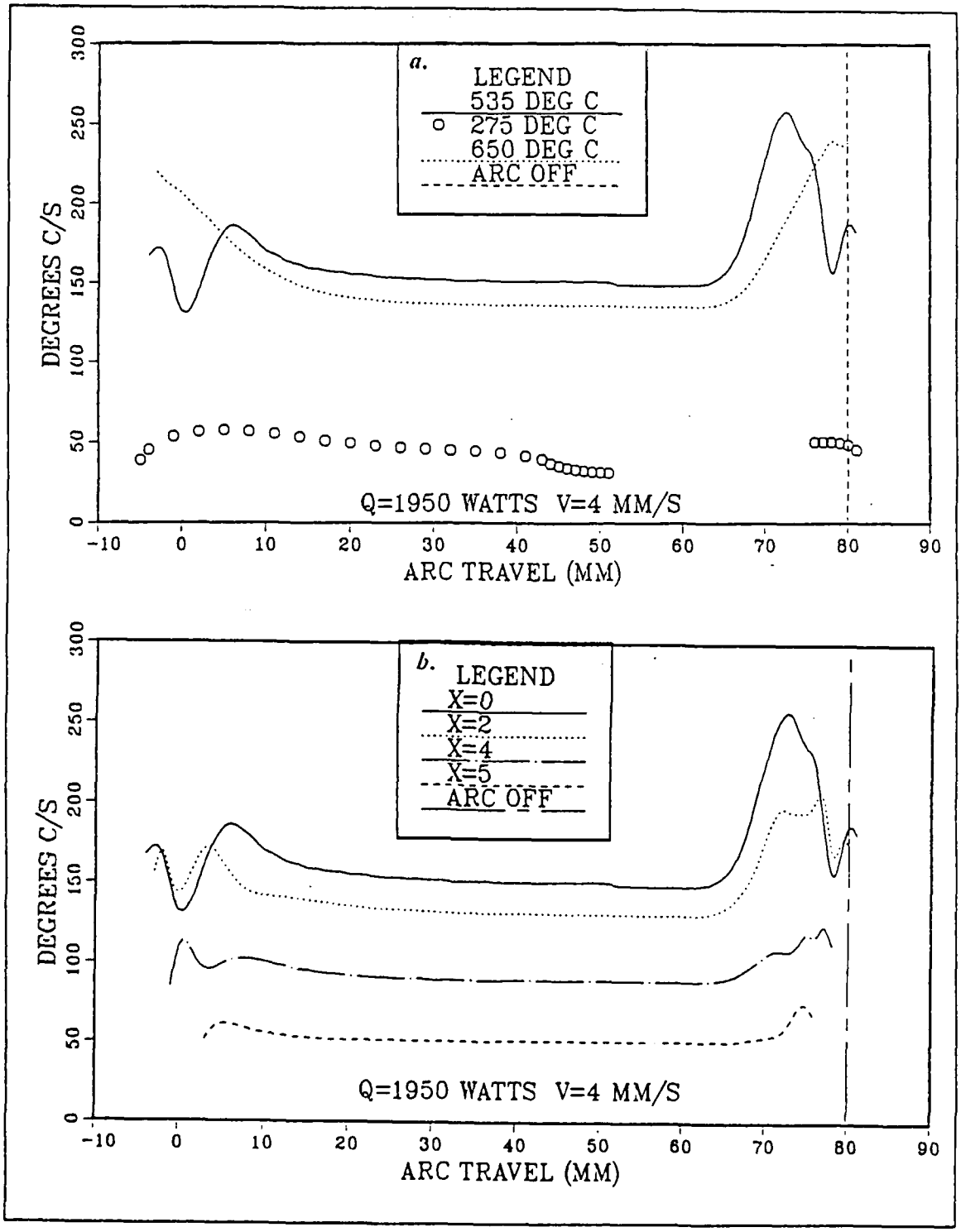


Figure 17. Baseline Cooling Rates at: a. Centerline b. 535 C

contour plot of the cooling rates around the arc path is shown for 650 degrees Celsius. This clearly shows the unacceptable cooling rates in the vicinity of the arc stop and start.

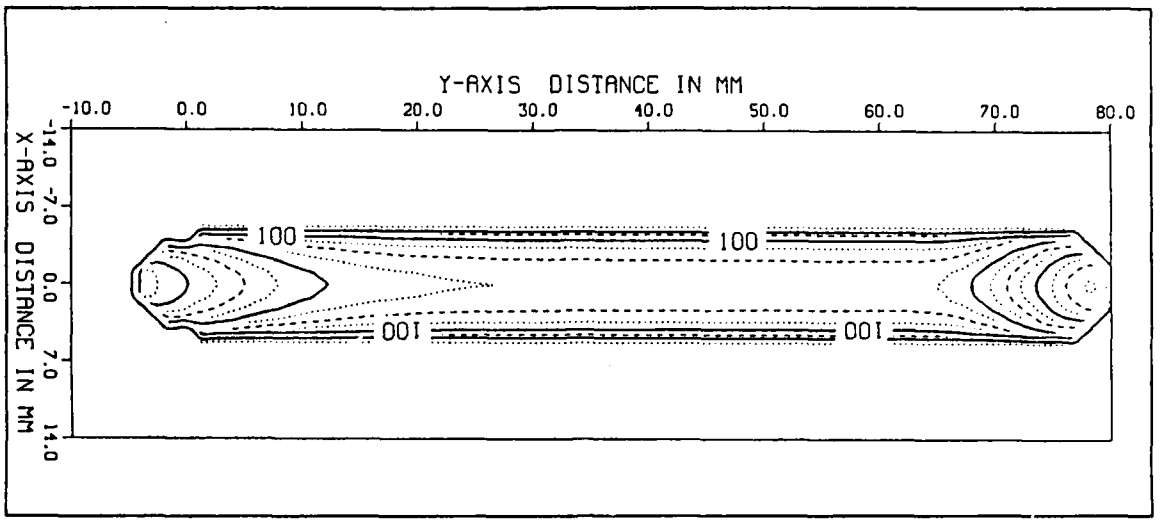


Figure 18. Baseline Contour Plot of Cooling Rate: The cooling rate is measured at 650 deg. C.

D. CONTROLLING COOLING RATES BY WELD SPEED AND POWER CHANGES

To control the cooling rate we can change both arc speed and arc velocity. Before randomly changing these parameters, the baseline profile was examined. It was reasoned that the high cooling rates at the start and stop of the welds were due to an energy deficiency. Therefore the best solution would be to augment the energy at the ends. Because the protective shield gases are still forming at the start of the weld it was decided to have the arc start slowly and accelerate to the normal operating speed. To test these ideas, a series of four different programs were tried using different combinations of torch speed and arc power.

1. Program One, Velocity Ramps

A constant acceleration was chosen for the first try. Since the quasi-steadystate took about ten millimeters of arc travel to reach, this distance was selected as the acceleration range. For the baseline this distance took 2.5 seconds, using constant acceleration it takes 5.0 seconds. Using this and constant arc power, the heat added over the first ten millimeters would be doubled, with more of the heat being added at the start of the arc. This would seem to nicely make up the energy deficiency. Similar reasoning was used for slowing the arc down at a constant acceleration prior to extinguishing the arc. This programming is shown in Figure 19.

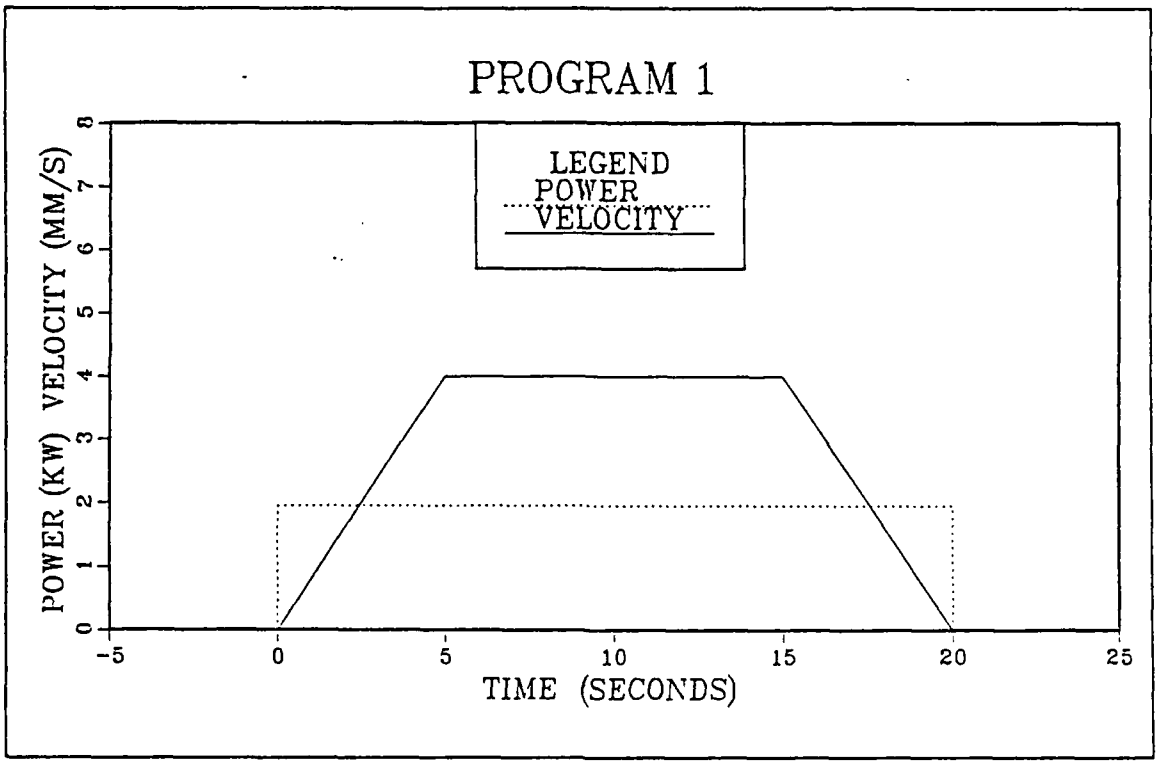


Figure 19. Velocity and Power Program One

The results of program one are shown in Figure 20. As desired, the cooling rates at the arc stop are now near the quasi-steadystate, the 650 degree Celsius cooling rate having been corrected the most. At first glance, it would appear that the drop in cooling rates at the very start and the trailing edge of the arc is a problem. This in in fact the correct response. Looking at the cooling rates off of the arc centerline at 535 degrees Celcius can help clarify this. The centerline cooling rate has almost dropped to the cooling rate five millimeters off of the centerline at the start. This occurs at a distance of minus five millimeters behind the torch starting position. Because the torch has never crossed this point, it is supposed to have a cooling rate similar to the heat affected zone. In Figure 21 this is clearly seen. The cooling rate contours with the programming are almost symmetrical around the torch path. At the start, the cooling rates at a fixed radial distance from the arc start location are the same.

Having the arc slow down at the end of the run to add the extra energy did not have the desired effect. The cooling rate at the point of arc slowdown dramatically dipped and the cooling rate at the point of arc off still steeply climbed and a slightly

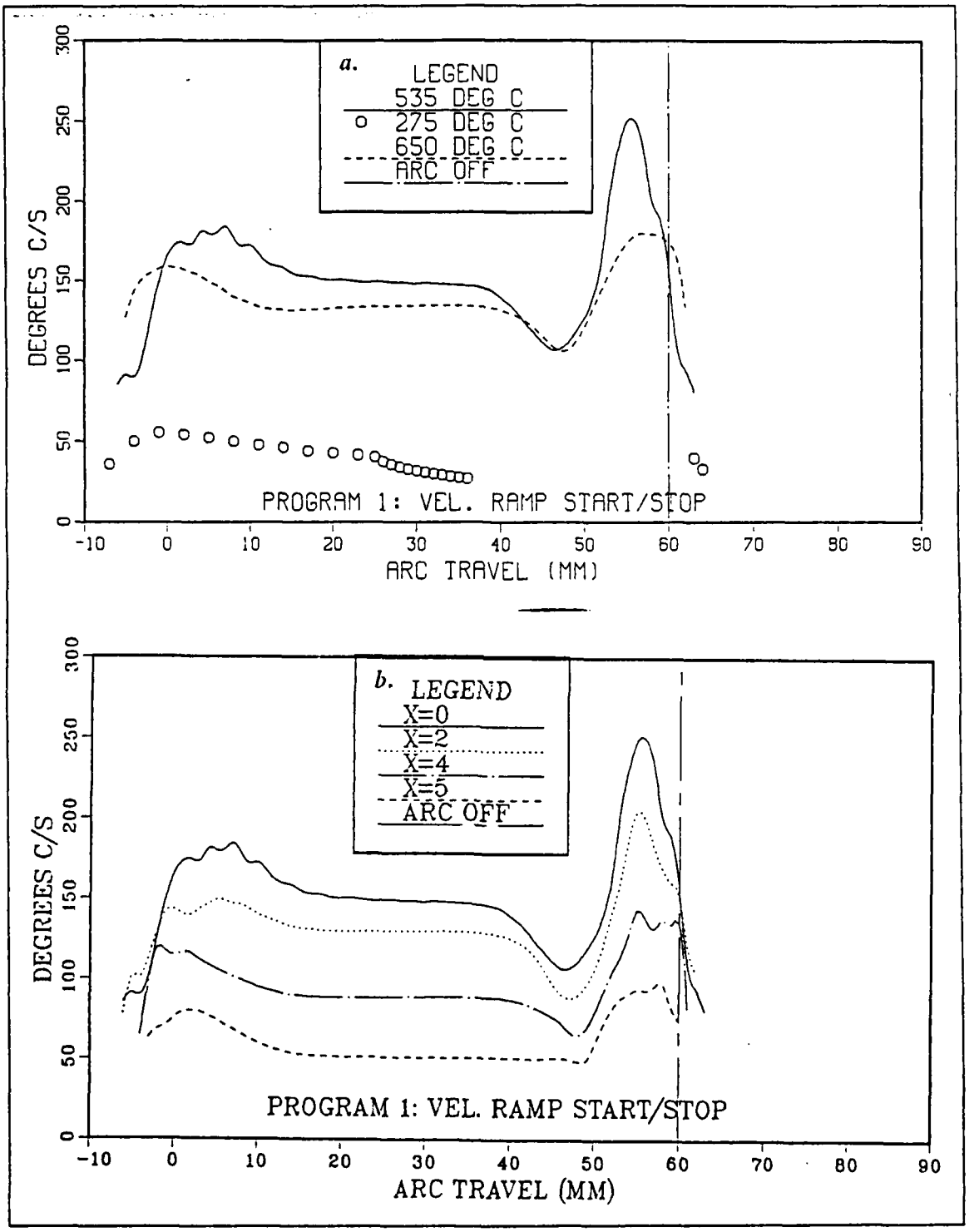


Figure 20. Program One Cooling Rates: a. Centerline b. 535 deg. C

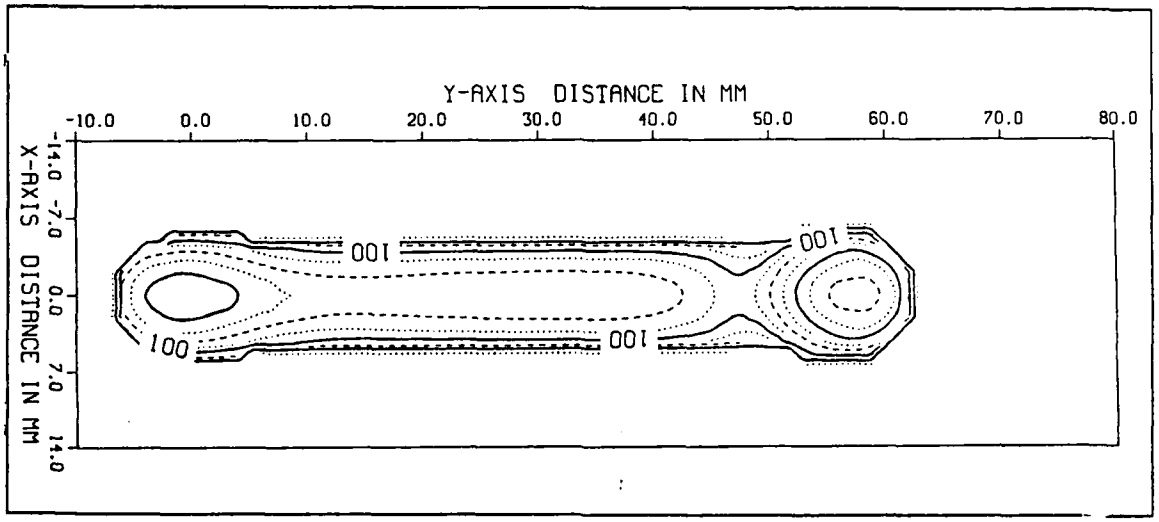


Figure 21. Program One Contour Plot of Cooling Rate: The cooling rate is measured at 650 deg. C.

higher cooling rate were obtained. The only benefit gained is that the cooling rates at stop also now show a radial symmetry. Reconsidering the problem, it is noted that the energy deficiency is not in the region near a quasi-steadystate profile, but effectively at the point the arc stops. Thus to correct this, maybe power, and not velocity programming should be used.

2. Program Two, Velocity Ramp on Start, Power Ramp on Stop

The velocity and power program two is shown in Figure 22. Here, once the arc has traveled the desired distance, the torch stops and the arc is left on. The arc power is linearly reduced to zero over a 2.5 second interval to make up the energy deficiency. Only one half of the estimated energy deficiency is made up in an attempt to control the cooling rate dip prior to turning off the arc. In (b) of Figure 22 it can be seen that the results are mixed. The dip in cooling rate prior to securing the arc is minimized and the 650 degree cooling rate is noticeably improved. However, the 535 degree Celsius cooling rate has slightly increased. This later effect is due to the way these cooling rates are measured; the 650 degree Celsius cooling rate is averaged over a 300 degree Celsius band, while the 535 degree Celsius cooling rate is averaged over a 30 degree band. This indicates that the ability to control a cooling rate depends on what temperatures it is taken over.

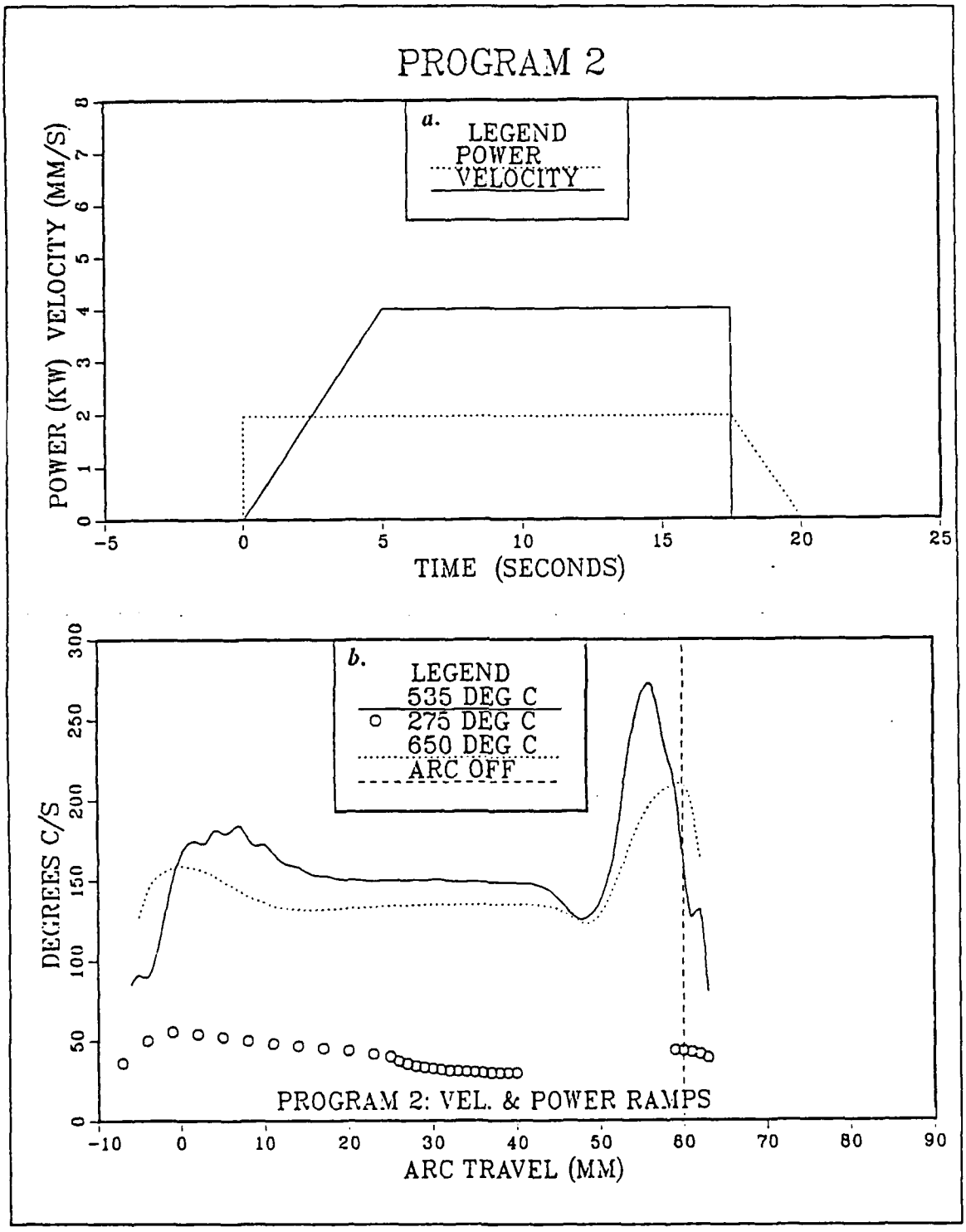


Figure 22. Program Two: a. Velocity & Power versus Time b. Cooling Rate

3. Program Three, Velocity Ramp and Holding Power

To further attempt control over the 650 degree Celsius cooling rate, the program shown in Figure 23 was tried. This starts as the others, but the power is held constant after the torch stops for an additional 2.5 seconds, vice ramping down as before. This would completely make up the estimated energy deficiency. As can be seen in part (b) of Figure 23 the 650 Celsius degree cooling rate was lowered even closer to quasi-steadystate when the arc is secured. As expected, the dip in cooling rate prior to stopping the arc has increased. The maximum 535 degree Celsius cooling rate is identical to the baseline.

4. Program Four, Constant Power to Velocity Ratio

Since in the quasi-steadystate, the cooling rate is directly rated to the ratio of arc power to torch velocity, it was conjectured that keeping this ratio constant while securing the arc would result in a constant cooling rate. This programming is shown in Figure 24. As can be seen in (b) of Figure 24 this is not the case. This was in fact the worst of the four programs with the largest excursions in both the 535 and 659 degree Celsius cooling rates.

E. CONCLUSIONS

It is possible to change the cooling rate during transients such as arc startup and shutdown. The best results were obtained for controlling the cooling rate when starting the arc. Since there is no thermal history affecting the diffusion of heat, significant changes can be affected as desired. The control of cooling rate on arc stop is more difficult. In this case there is a thermal history which will effect the cooling rates. The major noted problem, is that trying to make up an estimated energy deficiency caused a dip in the cooling rate prior to stopping the torch. This is because adding more energy at the end of the weld increases the weld pool size. Thus the region of the arc next to this enlarged pool has less low temperature material than the quasi-steadystate portion of the weld bead in which to diffuse heat. This results in lower cooling rates. Of the tests performed, program two seemed to be the best compromise.

Another important result is the idea of energy deficiency. By use of a simple energy balance and knowing the distance to quasi-steadystate it was possible to select a simple velocity and a power programming to keep the cooling rates near their quasi-steadystate values. This is important for the distance to quasi-steady state can be easily determined. Using this to develop a velocity and power program provides two major advantages. The first is that the run-on and run-off pieces can be reduced or eliminated. The second

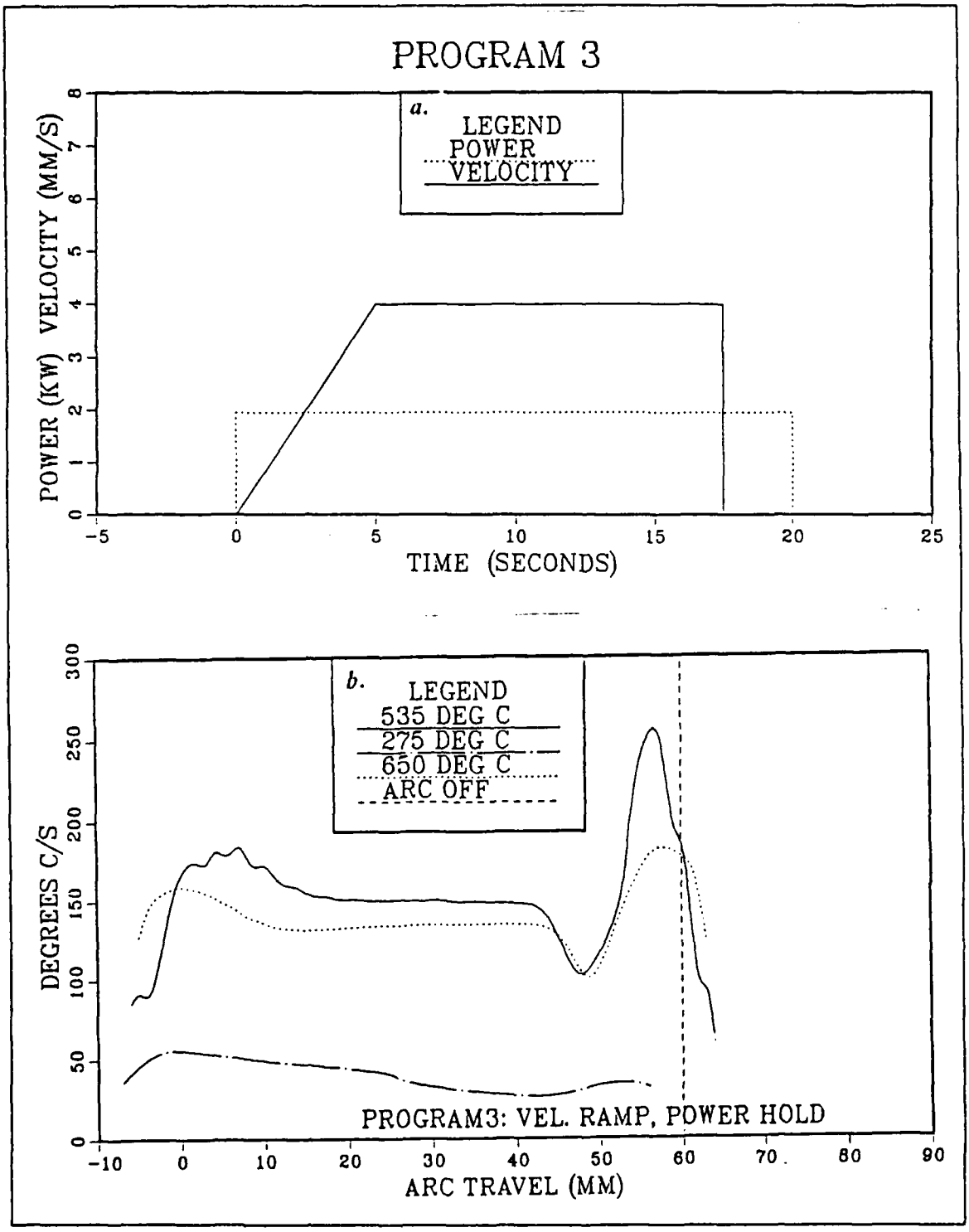


Figure 23. Program Three: a. Velocity & Power versus Time b. Cooling Rate

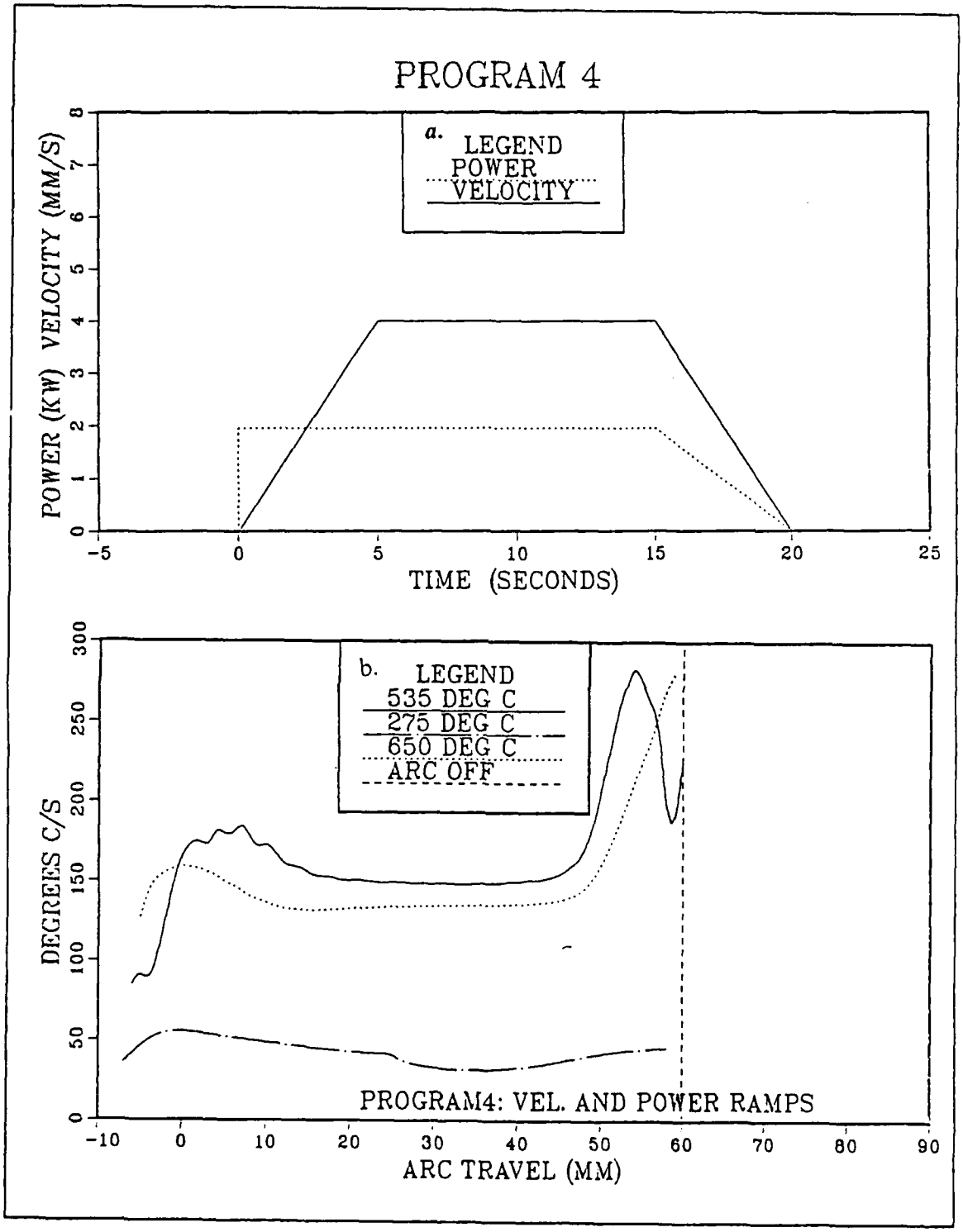


Figure 24. **Program Four: a. Velocity & Power versus Time b. Cooling Rate**

is that expert welders designed to control the cooling rate can use preprogrammed startup and shutdown algorithms, since during these transients any system will have inadequate feedback for effective control.

VII. THE INFRARED DIODE DETECTOR

A. INTRODUCTION

For many years welding was a skilled trade that was at times more art than science. As the study of the metallurgy of welding progressed, new welding techniques were invented. With the advent of modern robotic controls, welding tasks began to become automated. But, due to an incomplete understanding of all that occurs in the weld pool during the welding process, designing a closed loop control system of an automated welder is not a simple task.

Current automate welding systems attempt to eliminate as many variables as possible in the welding process, to simplify the task, and ensure better production rates. But

> ...these systems are incapable of correcting for perturbations that arise during the welding process. Control of the welding process requires the identification and monitoring of perturbations in a wide variety of parameters. The varied nature of these parameters and the large number of variables involved have thwarted previous attempts at closed loop control of the welding process.[Ref. 16: p. 799]

But even when care has been taken to remove the obvious variables, ensuring that there is a near perfect fit with no gaps between the pieces to be welded, for example, problems still arise.

The Edison Welding Institute reported in May, 1986, on a research project dealing with robotic vision in automated welding systems that

> ...accumulated variations in the motion and plate movement due to warpage and residual stress may cause the robot to miss the intended weld joint. The solution is to give the robot an eye and other sensors which "see" and "feel" when a correct weld is being made and can adjust itself when deviations occur.[Ref. 17]

This has been performed successfully by several researchers. Khan states that

> ...attempts to adapt intelligent vision systems to seam tracking and weld puddle control have been made. Most attempts of welding control have been adapted from computer-vision-based systems...recently infrared thermography has been used to monitor cooling rates in welds as a possible means of on-line-control of heat input. Although the above research has lead to increased knowledge of arc welding physics, an acceptable system for in-processing welding and quality control has not been developed. The development of improved sensors is required....[Ref. 16: p. 799]

A review of the literature found three different infrared thermography systems in use. These are the infrared scanning camera, the fiber optic spot sensor, and the solid

state video camera. Of these three, R. A. Morris, from David Taylor Research, reports that the solid state video camera is preferable to the nitrogen cooled scanning camera due to "...ease of data acquisition and reduced camera alignment problems."[Ref. 15] In addition the camera is less expensive to purchase and to operate. Given the requirements for a reliable infrared sensor, using infrared diodes was suggested as a low cost alternative. Because they are equivalent to a non-contact thermol couple, the diodes would provide much higher data acquisition rates, one area in which all present sensors are only marginal. This chapter explores the feasibility of using infrared diodes to monitor the weld process.

B. CHARACTERISTICS OF INDIUM ARSENIDE DIODES

The temperature band of interest for monitoring weld cooling rates is centered about 800 degrees Kelvin. Temperatures above this, at about 1300 degrees Kelvin, are useful for monitoring the heat affected zone to detect flaws and welding arc misalignment. The peak weld pool temperatures are around 2400 degrees Kelvin. A search of technical literature on infrared diodes located the Indium Arsenide series as having the best spectral response at room temperature. The response is shown in Figure 25, the peak radiative temperature corresponding to 2.0 microns and 4.0 microns are 2762 degrees Kelvin and 1381 degrees Kelvin, respectively. However at these low temperatures, the curves are in general very flat. That is, at low temperatures, the energy is spread out over a large frequency band. Thus, based on spectral response, the indium arsenide diodes are an acceptable sensor for detecting the temperature range of interest.

The indium arsenide diodes have a very small junction resistance, typically about 50 ohms. Thus, they are only operated in the short circuit mode. In the short circuit mode they provide linear current output versus light input for up to 9 order of magnitude. In addition, in the short circuit mode, the response of the diode is essentially independent of the diode temperature, removing the need for temperature compensation of the detector.[Ref. 18]

The indium arsenide diode purchased for testing was an ORIEL 71110, with a purchase price of $275.00. This diode has an active area of .79 mm^2 For testing it was mounted in ORIEL 7192 detector module, which is for unbiased operation. The detector provides a reverse current protection diode, since that maximum safe reverse voltage on an indium arsenide diode is one volt.

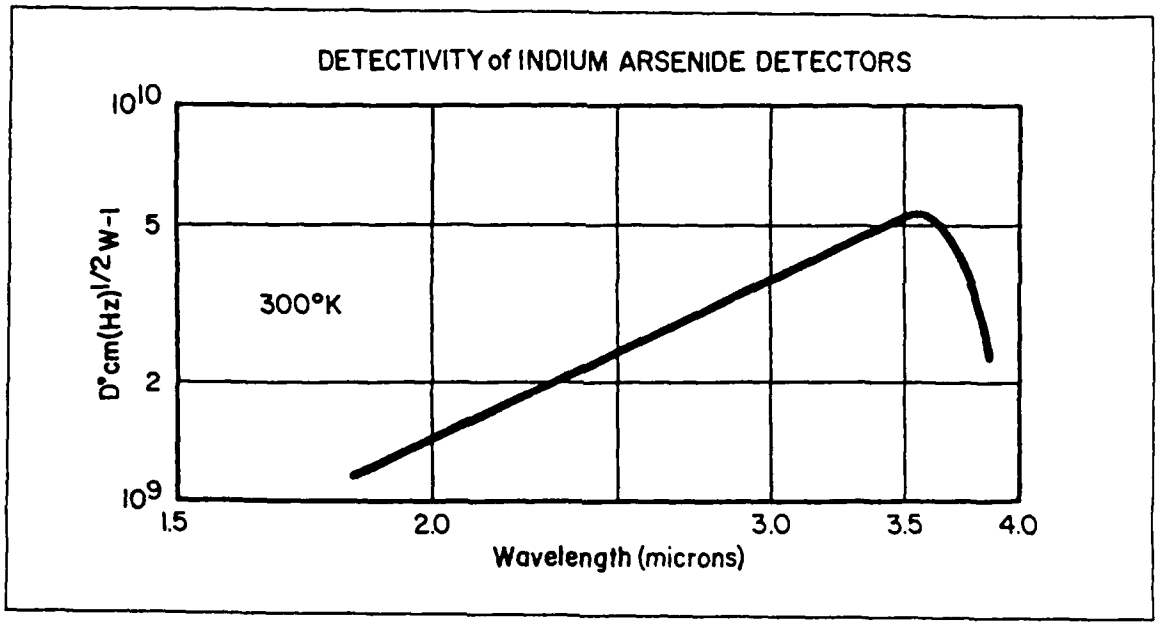


Figure 25. The Spectral Response of Indium Arsenide Detectors

C. THE DIODE DETECTOR ASSEMBLY

Other than the window of the diode, there is little control over the directionality. While advances have been made in infrared optics, a review of the literature showed that use of simple light pipes would be more than adequate for the type of focusing desired. Based on the results of computer modeling of flaws and misalignments, it was determined that a spatial averaging of the temperatures over an area of several square millimeters was a desirable feature of the sensor. Light pipes were selected since their use "...instead of a mirror and lens optics results in increased economy and simplicity of the apparatus."[Ref. 19]

The design of the light pipe was based on three considerations. To keep the diode out of the way of the welding torch. To minimize the signal loss due to the length of the light pipe. And to have an acceptable spatial resolution. After several trial light pipes, the one shown in Figure 27 was selected. A series of initial calibration runs were performed to find the best circuit. A zero bias, zero offset as shown in Figure 26 was initially used, with a gain resistance of Rg = 2K. Prior to calibration the offset output voltage was adjusted to a minimum by the trim pot, Ro. Since this particular circuit used an inexpensive 704 operational amplifier, the offset would drift as the circuit warmed up. Use of higher quality components would eliminate this problem. The drift in offset does not effect the measured output of the diode since it is always possible to

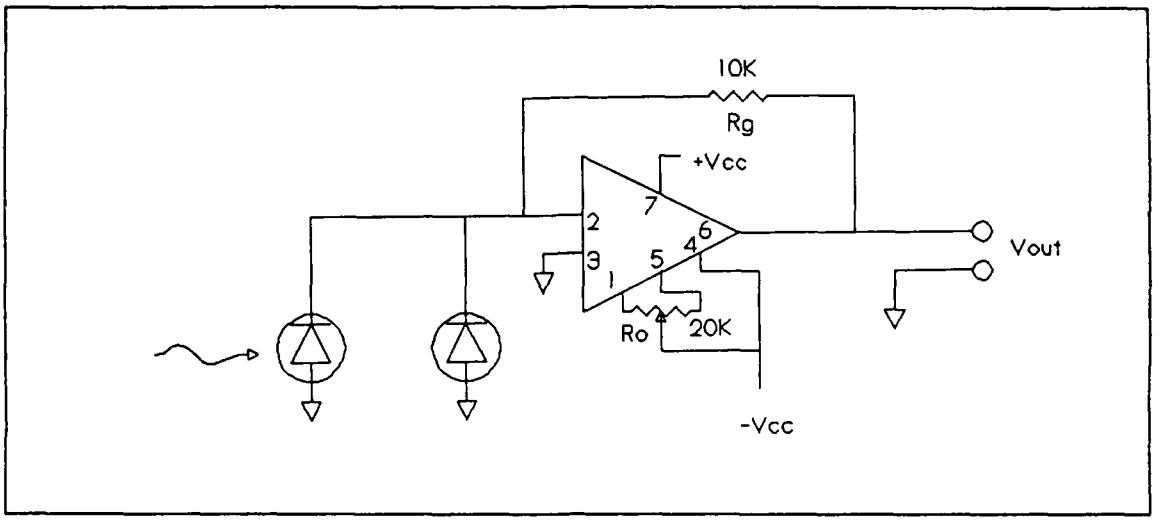


Figure 26. A Typical Zero Bias, Zero Offset Sensing Circuit: The circuit is shown using a 741 operational amplifier.

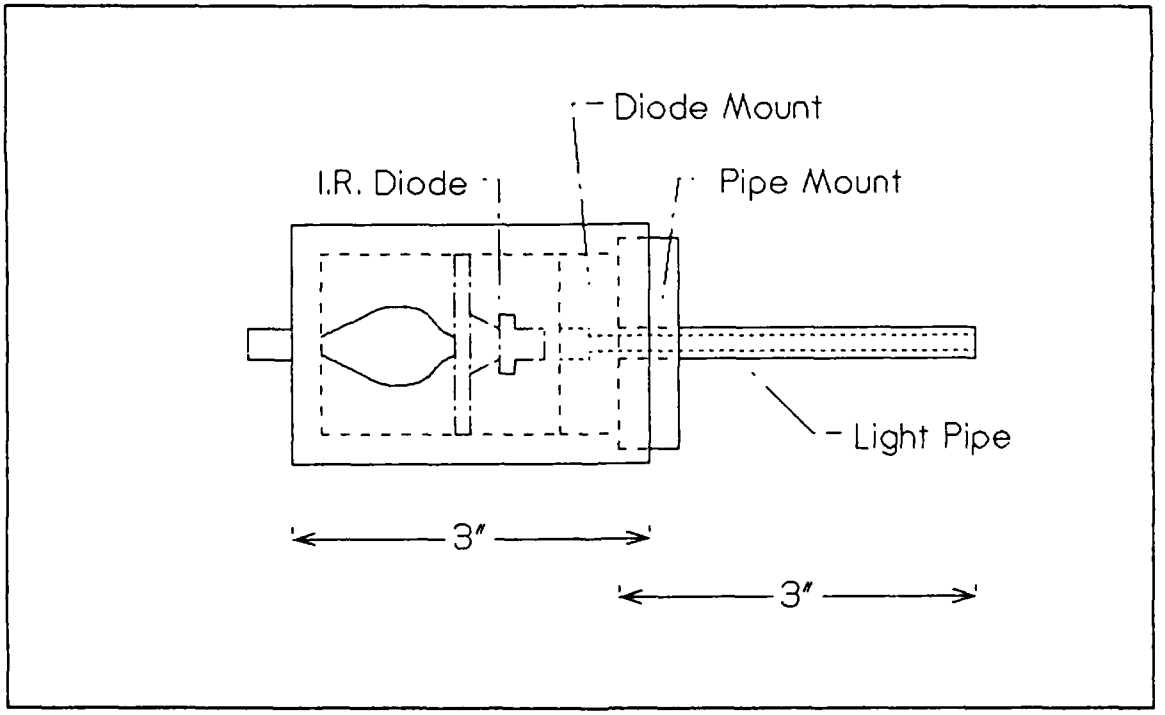


Figure 27. The Infrared Detector with Light Pipe: The stainless steel light pipe was press fitted into an aluminum holder, with the diode resting in its own mounting disk. Both are held into the Oriel 7192 detector with set screws.

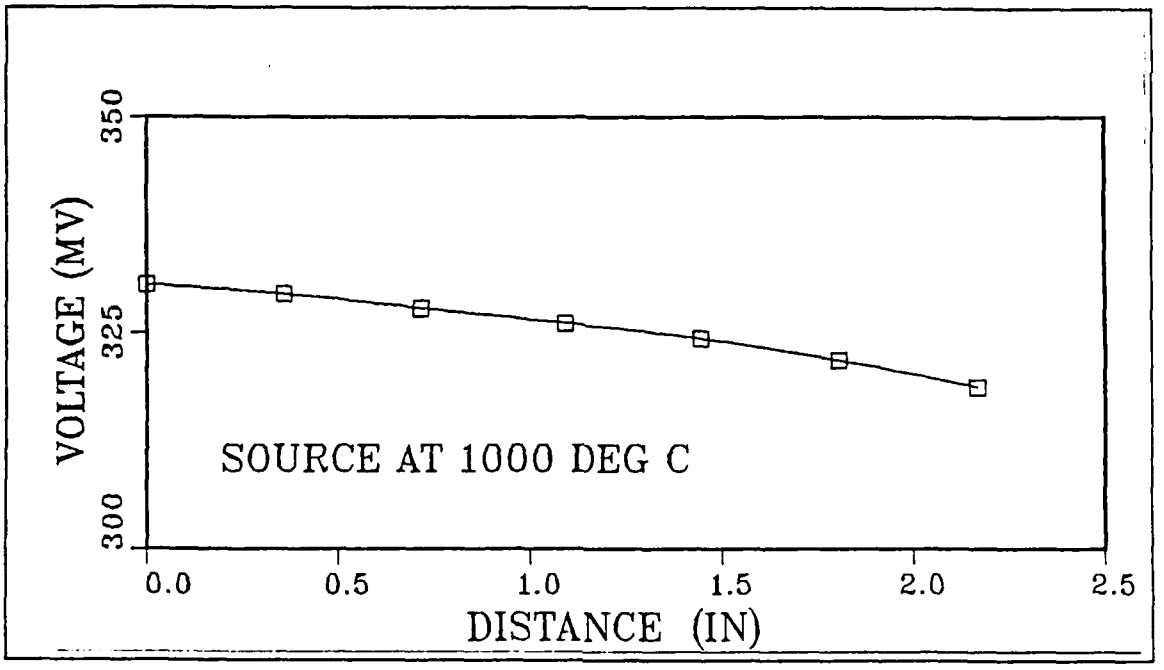


Figure 28. Sensitivity of a Light Pipe Detector to Distance: The calibration was done using the zero bias, zero offset sensing circuit.

measure the offset and subtract it away. What it does do, is limit the minimum temperature which can be detected, since the diode signal must be greater than the offset to be detected.

In Figure 28 is shown how the light pipe minimizes the sensitivity of the detector to minor changes in distance from the heat source. This is important since it makes alignment less critical and removes one more variable from the detector calibration. As expected, the light pipe did provide adequate focus so that little signal strength was lost for minor changes in distance from the source. A calibration curve is shown in Figure 29. The non-linearity does make the detector less suitable for measuring of cooling rates, however it is still suitable for seam tracking and flaw detection.

When the diode was zero biased, it was verified that the detector was insensitive to temperature changes. This was important, for in the biased condition the detector will rapidly lose signal strength as it heats up, due to its change in internal resistance. Testing showed that the diode would heat at as much as 0.5 degrees Celsius per minute. This heating was due to the radiant energy being absorbed by the diode. When the diode was tested in the open circuit mode, versus the short circuit mode, raising the temperature

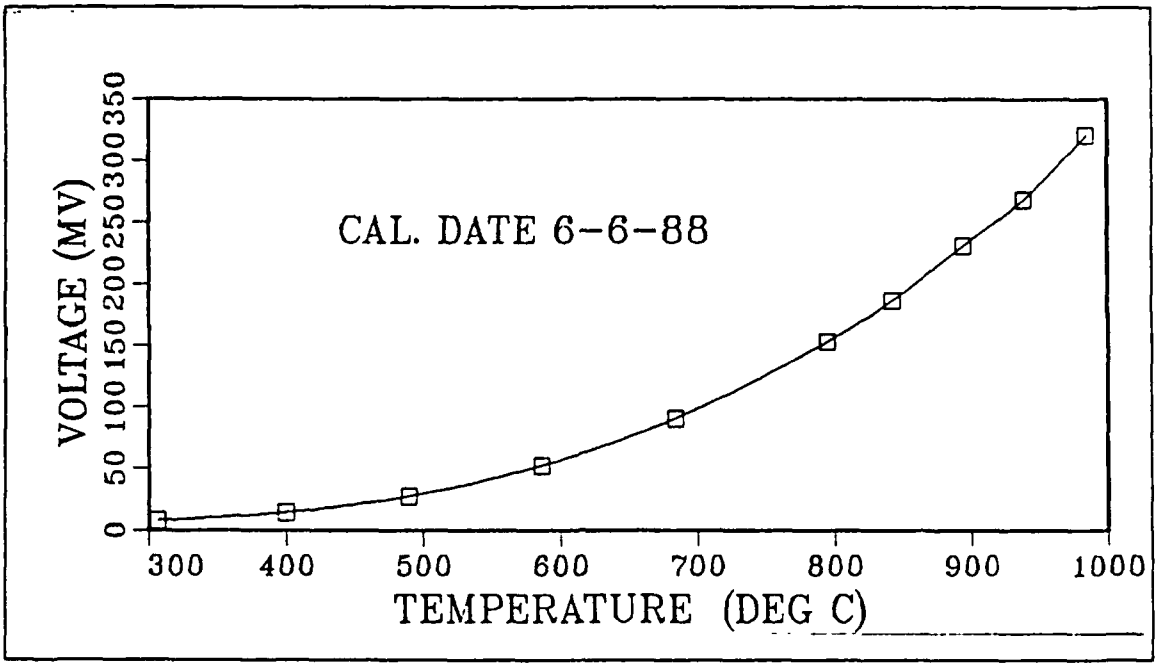


Figure 29. Typical Calibration Curve for an Indium Arsenide Diode: The calibration was done using the zero bias, zero offset sensing circuit. The diode dark voltage was five millivolts.

from 10 to 30 degrees C resulted in a 75 percent loss of signal. Thus, all further experimentation was done in the zero biased mode, that is short circuited.

D. CONCEPTUAL IMPLEMENTATION OF INDIUM ARSENIDE DIODES

The use of a simplified infrared vision scheme using lead sulfide photo conductance detectors was developed by Begin [Ref. 13: pp. 520-522]. This system was designed specifically for seam tracking and had a three dimensional capability. This allowed him to not only control the torch position relative to the seam, but also the torch height to control the arc length. The detection system he developed used a fiber optic cable to transmit the infrared energy from the work piece to his array of lead sulfide detectors. Though not mentioned in his article, this appears to be due to the bulkiness of the lead sulfide array and the need to keep these conductance detectors at a constant temperature to maintain calibration.

The indium arsenide diodes can directly replace the more complicated and expensive fiber optic cable and lead sulfide detectors. Since the diodes are only 1/8 inch in diameter, they can be directly mounted in to a light pipe assembly, such as shown in

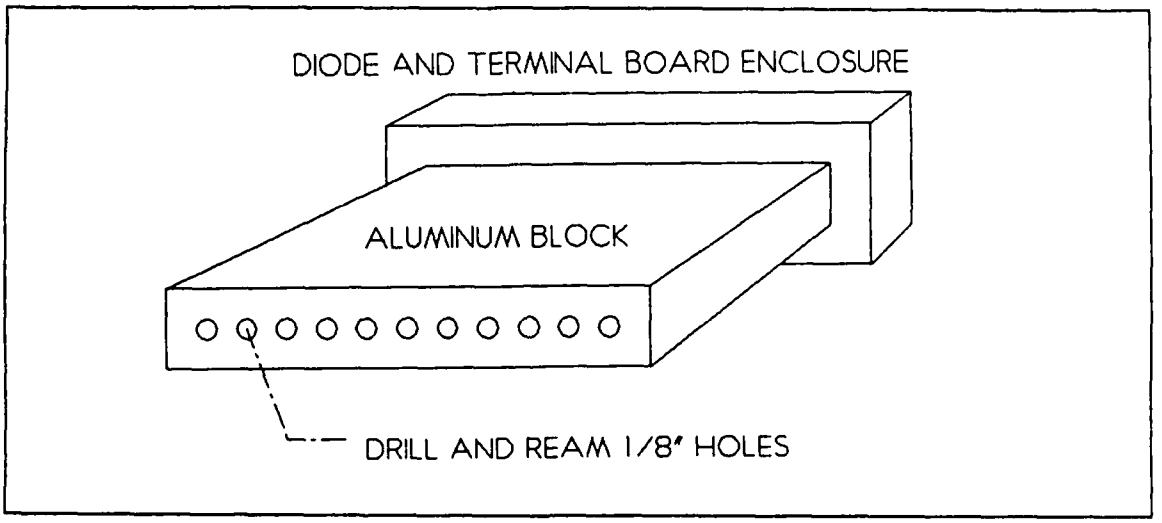


Figure 30. Proposed Configuration of a Light Pipe Array

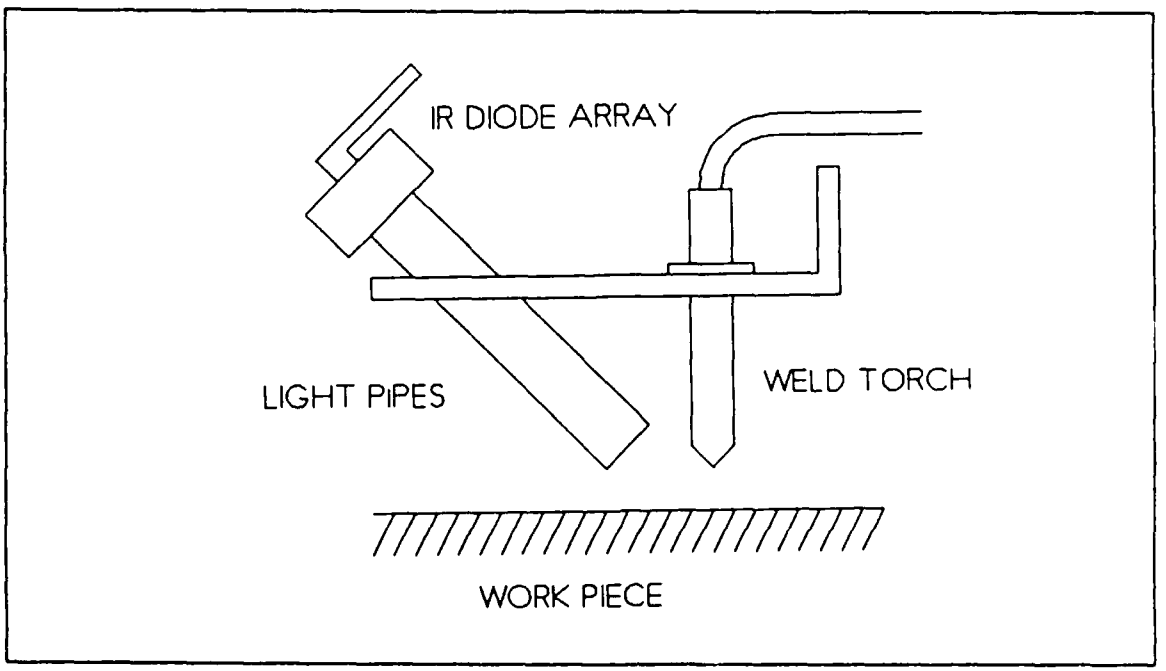


Figure 31. Proposed Mounting of a Light Pipe Detector Array

Figure 30, and still be in a package small enough to mount onto the weld torch. A possible installation of such an array is shown in Figure 31. Further more, it would be possible to mount individual diodes at exact positions of interest. This is useful, since no one location is ideally suited for taking every type of measurement. A list of possible locations are:

- The side of the arc: Looking for subsurface defects.
- The leading transverse profile: Looking for torch to seam misalignment.
- An angled leading transverse profile: Looking for torch height from work piece.
- Behind the arc: Measuring the cooling rate for material property control.

The spatial resolution of the detectors is a matter of some concern. Begin felt that the resolution was dependant only on the detection area of a given detector[Ref. 13: p. 521]. If a detector had a spatial resolution of 0.2 mm, then that was his spatial sensitivity. What he neglected to note is that the steep temperature gradients act as an amplifier. For example, the temperature gradient in the vicinity of the arc is of the order of 1000 degrees per inch. If the detector has a temperature resolution of 10 degrees, than the linear resolution would be 0.01 inch per inch of detector. So that a detector of 0.2 inches diameter would be able to detect an arc movement of 0.002 inches. Taking into account this amplification effect of the temperature gradient, it is possible to use fewer detectors with wider apertures. In addition, overlapping of detector windows, which is desirable, is now feasible. The use of the non-focusing light pipes is ideal for this type of coverage.

E. OPERATIONAL TEST OF THE INDIUM ARSENIDE DIODE

After calibration and sensitivity checks were completed the diode was taken to the welding laboratory to check its response in the presence of torch arc. An initial sensitivity check showed that the diode was not adversely affected by the torch arc. It also showed that the gain was too low for the temperature range of interest.

The diode circuit gain was increased by a factor of five, and the diode was recalibrated. It was mounted on a traverse to allow scanning a stationary arc thermal profile. When the circuit was energized, the offset was noted to have drifted since calibration to 0.06 volts from 0.01 volts. The torch arc was started with the diode directed two inches away. Upon attempting to read the diode, the diode had failed in the shorted condition. The cause of the diode failure has not been determined.

F. CONCLUSIONS

The use of the indium arsenide diodes or lead sulfide conductors is a suitable approach for a production environment infrared detector. While cameras and higher resolution vision schemes are useful for research in the laboratory, only that information required to control the welding process should be gathered for actual welding fabrication. Due to their small size, light weight, minimal temperature sensitivity and high

speed performance the diodes are more than adequate to gather the required thermal information. However, if it is determined that higher resolution is required, then the use of large numbers of these detectors would result in a rapid loss in any savings, and systems such as solid state video cameras should be considered. Also, further testing will be required to determine the suitability of the diodes for the production environment.

VIII. RECOMMENDATIONS

There are many areas in which this study can be continued. The model has proved itself adaptable to studying a wide variety of welding problems and can provide results with a reasonable computational cost. The indium arsenide diode has several features which make it a suitable detector in the production environment. Based on these results the following recommendations are made:

- Perform additional experimental verification of the model results.
- Verify the spatial dependence of the welding model.
- Investigate the effects of power level, torch velocity and material properties on flaw detection.
- Rewrite the cooling rate version of the model to take advantage of symmetry.
- Determine the transient distances for arc startup and shutdown.
- Develop a simplified vision system using indium arsenide diodes.

Some of the results of modeling have been verified by previous experimental work. This includes misalignments and surface flaws. The testing that is recommended, is the studying of subsurface flaws. The model predicted that the detectability of the flaw is dominated by its depth below the surface and it horizontal area. Since detectability is not significantly affected by the extension of the flaw into the plate, it should be possible to drill holes from the opposite side of the plate to various depths below the welding surface, and then these holes would approximate the flaws simulated in the computer model. The two major results desired would be to confirm that shallow flaws cause both a minor temperature rise and a larger temperature drop, and that deep flaws cause only a local temperature rise. To measure these changes, a scanning type detector vice a infrared camera will be required.

Experimental verification of the predictions on cooling rates could be done two ways. The first by use of an infrared camera to monitor temperatures and cooling rates during a weld sequence. The second is to section the material after the weld and analyze the heat affected zone's metallurgy. The last would be the more time consuming measurement, but would provide a detailed picture of the thermal history of the weld and heat affected zone. Using a simple power and velocity program scheme, as recommended by a simulation, a test weld could be performed and analyzed to determine the actual effectiveness of controlling the cooling rate.

The spatial dependence of the model is still of some concern. The selected one millimeter spacing was based on the experience of other researchers and simple two dimensional trials. When analyzing flaws, some boundary instabilities were observed, so that minor damped temperature oscillations occurred at the immediate location of the flaw. To establish this as the cause of the oscillation two choices are available. The boundary can be "softened" by increasing the flaws thermal conductivity from zero to an intermediate value, say 10. $w/m\,^{\circ}K$ The second is to change the grid size. Normally grid sizes are halved when checking spatial dependence. To do this would require eight times as many nodes, and due to the time step limitation, taking one fourth the time step. This would cause the model to run 32 times longer! Based on this, it is recommended that the grid size be reduced by a larger fraction, say 3/4. In this case the model would only run 4.2 times longer for the same simulation.

The detectability of flaws was limited to the study of the effect of flaw location and geometry. The effect of power was studied over a small range, and in general no correlation was found between power changes and the relative changes of temperature from quasi-steadystate. A further study of the effects of power, as well as torch velocity and material properties should be pursued using the model. Based on this research it should be possible to generate a more general correlation between a given flaw and its detectability including these variables.

The program used to simulate weld cooling rate, WELDC, does not invoke symmetry around the arc path axis to halve the number of nodes. Because the problem being study was symmetrical, this could have been done. While this is a significant rewrite of the code, it may be required if more realistic welding conditions of higher arc power and welding speeds are desired. These later will require extending at least the medium grid nodes behind the arc even further, lengthening a program which presently requires about one hour of CPU for every five seconds of simulation. By using symmetry, the computational cost could be halved.

One of the interesting results of the cooling rates experiment was the importance of the transient length during startup and shutdown. By simply knowing this distance for a given set of welding parameters it was possible to develop a program to control the cooling rates. Being able to predict these lengths would then allow programming for any set of welding conditions. It may be possible to determine these distances analytically, and if not, repeated computer experimentation could provide a basis for prediction.

Due to the speed of the model using Runge-Kutta, it may now be possible to expand the code to include weld pool simulation. Normally, this is not possible because the

dynamics of the weld pool require more nodes than are available in a model. The simplest case would be the development of a stationary weld pool. This would allow eliminating the coarse zone and shrinking the medium zone to reduce the number of nodes. It would then be possible to increase the number of nodes in the weld pool and still have a system that could be solved in a reasonable amount of time. This is particularly true when using a mini-VAX, which could be used, since the explicit technique uses very little memory compared to an implicit technique. Thus a problem of interest could be set into a dedicated mini-VAX and left to run for a day, at essentially no cost to the user.

The final recommendation is that a simplified vision system be prototyped using indium arsenide diodes. To successfully use infrared techniques in production welding requires that the detectors be as simple to use and rugged as possible. These diodes are relatively inexpensive and can be easily mounted with the welding torch. They require no special support equipment, and with a simple zero bias detection circuit can be directly fed into any analog-to-digital converter for use by an expert welding system. Further testing is required to determine their suitability for the production environment.

APPENDIX A. DETAILED EXPLANATION OF THE WELDING COMPUTER MODEL

A. EXPLANATION OF THE SOURCE CODE

The program has four major parts. The first is the main program called WELD. This controls input and output, adds the energy from the arc, uses Runge-Kutta to solve the semi-discrete difference equations and repositions the grids relative to one another. The other three are the subroutines that contain the finite difference equations for each grid zone; the fine, medium and coarse. These subroutines are used by the Runge-Kutta solver to find the temperatures at the next time step. A listing of the the variables used and their definitions is found in Table 2 below. The only inconsistency is in the use of the arrays in the subroutines. In the subroutines the input arrays are referred to as A, B or C instead of AIN, BIN or CIN. The discussion that follows refers to the source listings.

Table 2. VARIABLES USED AND THEIR DEFINITIONS

Variable	Definition
A(21,36)	Array of coarse grid temperatures in degrees Kelvin
AIN(21,36)	Array of coarse temperatures for input of the next Runge-Kutta step
AOUT(21,36)	Array of coarse temperature changes on output of the Runge-Kutta step
ASUM(21,36)	Array of coarse temperature change sums used by Runge-Kutta
B(27,27)	Array of medium grid temperatures in degrees Kelvin
BIN(27,27)	Array of medium temperatures for input of the next Runge-Kutta step
BOUT(27,27)	Array of medium temperature changes on output of the Runge-Kutta step
BSUM(27,27)	Array of medium temperature change sums used by Runge-Kutta
C(27,27)	Array of fine grid enthalpy in Joules per cubic meter
CIN(27,27)	Array of fine temperatures for input of the next Runge-Kutta step
COUT(27,27)	Array of fine enthalpy changes on output of the Runge-Kutta step
CSUM(27,27)	Array of fine enthalpy change sums used by Runge-Kutta
BI(4)	The Biot numbers for different grids

FO(3)	The Fourier numbers for different grids
AMP	The current of the arc in amperes
BSTEP	Used to count the number of times the fine grid has shifted, when it reaches three, the medium grid shifts and it is set to zero.
DELT	The incremental time step in seconds
DIS	The distance the arc has traveled in millimeters
EFF	The efficiency of the arc
FINI	Time in seconds to run simulation
FKB	The fixed thermal conductivity used by the medium grid in W mK
M	Counter of which time step is being solved
N	Counter of total number of steps taken, will be different from M only if problem has been restarted.
NB	Pointer which locates the medium grid relative to the coarse grid
NC	Pointer which locates the fine grid relative to the medium grid
NDIV	The number of divisions (or time steps) to be taken
OUT	Compared with time to decide when to output data, is then increased by 0.5 seconds
Q	The amount of enthalpy change of a given control volume for each time step in Joules per cubic meter
QDOT	The rate of energy input from the arc in Watts
STEP	Used to compare the distance the arc has traveled, when they are equal the fine grid shifts and it is increased by three units
SUM	The spatial weighting factor for arc density, varies with arc position
TIME	The time since the start of the simulation in seconds
TINF	The temperature of the surroundings in degrees Kelvin
TINF4	The temperature of the surroundings raised to the fourth power
VOLT	The voltage of the arc in volts
XARC	Position of the arc on the fine grid along X (i) direction in millimeters
XVEL	The velocity of the arc in the X (i) direction in mm s
YARC	Position of the arc on the fine grid along Y (j) direction in millimeters
YVEL	The velocity of the arc in the Y (j) direction in mm's

1. Main Program Weld Fortran

The main program, WELD, is divided into five sections, plus three functions. These are input and setup, adding heat from the arc, the Runge-Kutta solver, running

output, and final output plus setup for a restart. The three functions are for determining the thermal properties of the fine grid. These are thermal conductivity as a function of temperature, and converting to and from temperature and enthalpy.

a. Input and Setup of a Problem

The first section of the main program deals with input for setting up the problem. Once the initial validation was complete, the program was run as a batch job. Thus prior to each run the program is recompiled with the appropriate fixed conditions. For each run the length of the simulation is specified using FINI. If a new data run was commencing, the GOTO 100 line is commented out and the initial conditions are set and the output files are opened. If this is a continuation of a previous run, then the program jumps to statement 100 where the data from the previous run is loaded and the output files are prepared. In general, the continuous output files SURF and CUT are repositioned to their ends to allow appending additional data. The initialization of parameters is skipped by reentering at statement 200. Then the material properties are defined and the program enters the main processing loop.

b. Adding Heat from the Arc

The first step of the main processing loop calculates the energy added by the arc. It first calculates a volume weighting factor, SUM based on the arc's present X-Y location. This is necessary because the amount of energy added to a nodal point is based on its distance from the arc center, which changes each time step. Thus, to keep the energy added each time step the same, the spatial distribution is normalized by the use of SUM. SUM and the actual energy input for this time step, Q, to determine the change in enthalpy of the nodes under the arc in the fine grid. Obtaining the position of the arc relative to the fine grid is done using the variables XARC and YARC, where XARC is usually constant at 14 and YARC is as defined in equation A.1.

$$YARC = VEL \times TIME + 73 - 9NB - 3NC \qquad (A.1)$$

YARC uses the two pointers, NB and NC, to calculate the relative position of the arc on the fine grid from the fixed reference of the coarse grid. The value of 73 is based on an initial NB = 3, NC = 10 and the arc at position 16 on the fine grid. YARC is in millimeters.

The Gaussian power distribution of the arc is done using equation A.2, which is shown for the case of a hemisphere of radius four millimeters, that is where a = b = c = 4 mm.

$$Q(x,y,z) = \frac{C_1}{abc}\exp\left[-1.7\left(\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}\right)\right] \qquad (A.2)$$
$$= C_1 \exp\left[-0.10625(x^2+y^2+z^2)\right]$$

The value of the change in enthalpy at each node is approximated by taking the value at the nodal point and assuming it is constant over the control volume of the node. To insure a proper energy balance, a weighting factor is calculated by summing the values of Q at each node. SUM is then used to normalize C_1, where C_1 = Q/SUM. Hence at each time step, the same total change in enthalpy occurs. When double elipsoids are used, that is the power distribution is not symetrical about the center of the arc, then it is necessary to use the first form of the Gaussian power distribution.

c. Runge-Kutta Solver

The Runge-Kutta solver requires temperatures, so the first step is to convert the fine grid enthalpy to temperature. The function T(h) performs this conversion and places the temperatures into the array CIN. The basic form of fourth order Runge-Kutta is shown in equation A.3,

$$\begin{aligned} K1^n &= f(T^n) \\ K2^n &= f\left(T^n + \frac{K1^n}{2}\right) \\ K3^n &= f\left(T^n + \frac{K2^n}{2}\right) \\ K4^n &= f(T^n + K3^n) \\ T^{n+1} &= T^n + \frac{(K1^n + 2(K2^n + K3^n) + K4^n)}{6} \end{aligned} \qquad (A.3)$$

where $f(T)$ is the semi-discrete form of the difference equation. By combining terms between steps, it is possible to use only three arrays per grid, vice the five listed above. The input to $f(T)$ is the IN array and the K is the OUT array. After each step, a new IN array is calculated by adding the OUT array values, and the OUT array values are added to the SUM array. For the fine grid, the COUT array is in enthalpy, so that the new CIN values are found using the function T(H). After the fourth Runge-Kutta step, the A, B and C arrays are updated to the next time step by adding one sixth of the ASUM, BSUM and CSUM arrays, respectively.

d. Moving the grids

The variable STEP is used to decide when to move the fine grid. If the distance traveled equals or exceeds STEP then it is time to move the fine grid. In this case STEP is increased by three millimeters and the counter for moving the medium grid, BSTEP, is incremented one. Moving the fine grid is done in three steps. First the trailing edge of the grid will be now in the medium grid. These are all fine nodes with a j index of 1, 2 or 3. There is one medium grid node for each 27 fine grid nodes (18 fine nodes on the surface). The enthalpy for the fine grid nodes are averaged together and converted to a temperature which is assigned to the equivalent medium grid node. Care is taken for the surface nodes, those with a k index of 1, because they only have half the volume of a regular node. The second step is to shift all the values of the fine grid. Those nodes with a j index of 1, 2 or 3 have already been accounted for, so the whole fine array is shifted -3 in the j index. The last step is to assign values to the leading edge nodes, those with a j index of 25, 26 or 27. They are all assigned an enthalpy equivalent to the temperature of the medium grid node they will now be occupying. This conversion is done using the function H(T). Upon completion, the pointer NC is incremented one to indicate the new location of the fine grid.

In the case that BSTEP is now equal to three, it is time to shift the medium grid in the coarse grid. BSTEP is set equal to zero and the grid is again shifted in three steps. The first step is to average the trailing nodal points and assign them to the equivalent coarse grid nodal point. There are 90 medium nodal points for each coarse nodal point. Those nodes with a j index of 1, 2 or 3 are averaged together, taking care to note that the surface nodes, those with a k index of 1 or 10 have only half the volume of a regular node. The second step is to shift all of the nodes -3 in the j index direction, since those nodes with a j index of 1, 2 or 3 are already accounted for. The last step is to assign the temperature from the equivalent coarse node to the leading edge nodes, those with a j index of 25, 26 or 27. Upon completion, the pointer NB is incremented one to indicate the new location of the medium grid and NC is reset to 10 to indicate the new location of the fine grid.

e. Running Output

During the running of the program, snapshots are taken of the temperature every half second and saved. The data is saved in two files, SURF and CUT, which are then used to generate three dimensional plots and contour plots of the temperatures. The data saved are the surface temperatures of the fine grid, the surface temperature of the medium grid and the X-Z profile of the fine grid directly under the arc. For the fine

grid output the function T(H) is used to convert the enthalpy to temperature. For the medium grid, the nodal points occupied by the fine grid do not normally have a value assigned. For continuity, the enthalpy of the fine grid nodes are averaged together, converted to temperature and assigned to their equivalent medium grid node. This is identical to the process used by the grid shifting section of the program.

f. Final Output and Setup for Restart

Upon completion of the program the time and the final temperatures of the fine, medium and coarse grid are placed in a file called FINAL. The regions in the medium and coarse grid which are occupied by another grid are filled in by averaging the values from the finer occupying grid, as is done during grid shifting. This file is used to generate three dimensional and contour plots of the temperature profiles. A second file, RESTAR, is used for restarting the problem. It saves the parameters necessary to restart the problem as well as the three arrays; A, B and C.

g. Functions

The WELD program uses three function, FK, T and H. The first is FK, which models thermal conductivity as a piece-wise linear function of temperature, which is discontinuous only at melting. The second is T, which models the temperature as a piece-wise continuous linear function of enthalpy. The third is H, which models the enthalpy as a piece-wise continuous linear function of temperature. The values are the same as used by Goldak [Ref. 5: pp. 587-600].

2. Subroutine FIN Fortran

This subroutine contains the semi-discrete form of the explicit finite difference equation for the fine grid. The parameters for this grid are listed in Table 3. The subroutine is broken into nine sections, each dealing with a different type of node. These nodes are shown in Figure 32, where the letters in the figure correspond to the following sections. The program makes use of a function GK, which is shown in equation A.4, to improve the readability of the code. The function GK takes the harmonic mean of the thermal conductivities and multiplies it with the temperature difference for each difference pair. When this is done for all adjacent nodes, the change in enthalpy at that node is known. The enthalpy form of the equation takes into account the variation of the heat capacity with temperature.

$$GK(T_1, T_2) = \frac{2FK(T_1)(T_1 - T_2)}{FK(T_1) + FK(T_2)} \qquad (A.4)$$

Table 3. PARAMETERS DEFINING THE FINE GRID

Dimensions: 27mm x 27mm x 7.5mm Node Volume: 1mm x 1mm x 1mm Array: C(27,27,8)
The following constants are used: $FO(3) = 1.0x10^{6}\ \Delta t\ s/m^2$ $BI(3) = 9.399x10^{-5}\ \Delta t\ J/m^3K^4$ $BI(4) = 5.0x10^{5}\ \Delta t\ J/m^3K$ $FKB = 26.5\ W/mK$ $Kb = 53.0\ W/mK$ $Cp = 4.0x10^{6}\ J/m^3K$ $h = 25.0\ W/m^2K$ $\varepsilon = 0.82$ $\sigma = 5.67x10^{-8}\ W/m^2K^4$

The function FK(T) returns the thermal conductivity for the input temperature.

Three of the constants in Table 3 require a little explanation. The first, FO(3), is like the Fourier number, and is defined by $FO(3) = \Delta t/\Delta x^2$. The second is an equivalent Biot number for radiation from the surfaces. This is defined by $BI(3) = \varepsilon\sigma\Delta t/\Delta z$. The last is the equivalent Biot number for convection from the surfaces. This is defined by $BI(4) = h\Delta t/\Delta z$. Note that the Biot numbers are with respect to Δz, not Δx and that Δz is one half of Δx for a surface node. These two equivalent Biot numbers are the product of the normal Fourier and Biot numbers.

Interfacing between grids requires care. As discussed in Chapter 2 the nodal spacing controls the difference equation. The distance to a medium node is twice that of normal fine node spacing. Because the change in enthalpy at a node is related to $1/\Delta x$ this reduces the effect by one half. This is why FKB is one half of Kb. The second difficulty in interfacing is keeping track of the association of fine nodal points to adjacent medium grid nodal points. In the subroutine all medium array indexes are combined with a B to indicate they are B array subscripts.

a. The Interior Nodes

The equation for the interior node is the simplest. This equation is valid over the following range: i = 2 to 26, j = 2 to 26 and k = 2 to 7.

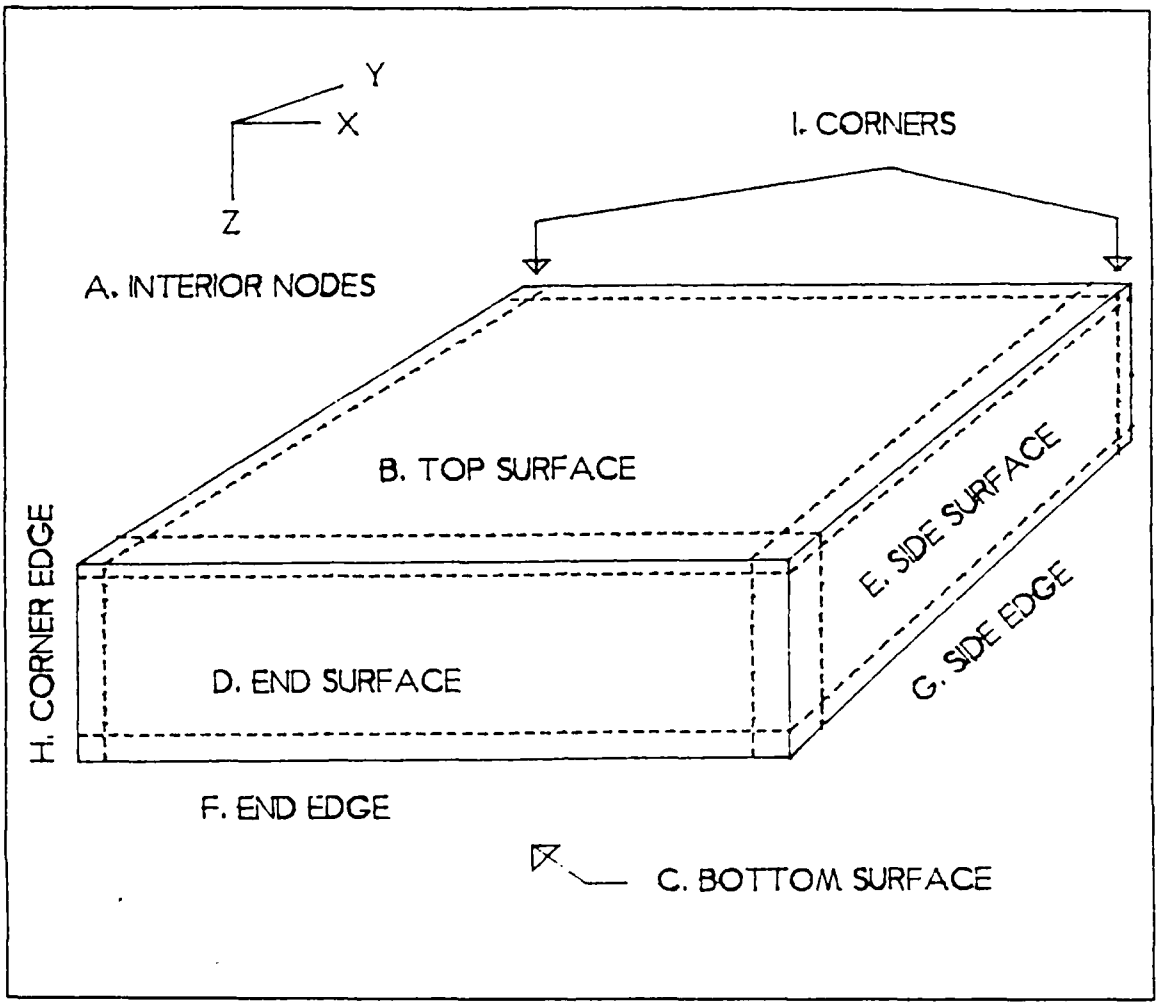


Figure 32. The Fine Grid Node Types: The letters coresspond to the sections of the subroutine FIN.

b. The Top Surface Nodes

The difference equation for the top surface nodes is similar to an interior node but it has a convection term and a radiation term. Because Δz is one half of Δx, the difference of the k index has twice the effect of the other differences. This equation is valid over the following range: i = 2 to 26, j = 2 to 26 and k = 1.

c. The Bottom Surface Nodes (Medium Interface)

The difference between this equation and the interior node equation is that the +k index term is replaced by a difference term with the medium grid. FKB takes care of the difference in node spacing between the fine and medium zone. This equation

is valid over the following range: i = 2 to 26, j = 2 to 26, k = 8, IB = 10 to 18, JB = NC to NC+8, KB = 4.

d. The End Faces (Medium Interface)

The end faces are similar to the bottom nodes. Depending on the face, either the +j or -j index term has been replaced by a medium grid difference term. As before, care was taken in matching the fine to the medium nodal points. These equations are valid over the range: i = 2 to 26, j = 1 or 27, k = 2 to 7, IB = 10 to 18, JB = NC-1 or NC+9, KB = 1 to 3.

e. The Side Faces (Medium Interface)

The side faces are similar to the end faces, with either the +i or -i index term being replace by a medium grid difference term. These equations are valid over the range: i = 1 or 27, j = 2 to 26, k = 2 to 7, IB = 9 or 19, JB = NC to NC+8, KB = 1 to 3.

f. The Top and Bottom End Edges (Medium Interface)

This is a combination of dealing with grid interfaces and surfaces. The top equation has either the +j or -j index term replaced by a medium grid difference term and the -k index term is replaced by the two surface terms for convection and radiation. The bottom equation has either the +j or -j index term and the +k index term both replaced by a medium grid difference term. These equations are valid over the range: i = 2 to 26, j = 1 or 27, k = 1 or 8, IB = 10 to 18. For the top edges JB = NC-1 or NC+9 and KB = 1. For the bottom edges; one term will be JB = NC-1 or NC+9 and KB = 3, for the other term JB = NC or NC+8 and KB = 4.

g. The Top and Bottom Side Edges (Medium Interface)

This is the same as the end edges. The top equation has either the +i or -i index term replaced by a medium grid difference term and the -k index term is replaced by the two surface terms for convection and radiation. The bottom equation has either the +i or - j index term and the +k index term both replaced by a medium grid difference term. These equations are valid over the range: i = 1 or 27, j = 2 to 26, k = 1 or 8, JB = NC to NC+8. For the top edges IB = 9 or 19 and KB = 1. For the bottom edges; one term will be IB = 9 or 19 and KB = 3, for the other term IB = 10 or 18 and KB = 4.

h. The Corner Edges (Medium Interface)

These have no surfaces nodes, just interfaces with the medium grid. The equations have both the +i or -i index term and the +j or -j index term replaced by a medium grid difference term. These equations are valid over the range: i = 1 or 27, j

= 1 or 27, k = 2 to 7, KB = 1 to 3. One term will be IB = 9 or 19 and JB = NC or NC+8 and the other term will be IB = 10 or 18 and JB = NC-1 or NC+9.

i. The Top and Bottom Corners (Medium Interface)

Each of these eight equations is unique. They take into account a surface term for those on the top and an interface term for those on the bottom. In addition they each have two interface terms in the +i or -i index and the +j or -j index. The fine grid nodal points that are corners are: (1,1,1) (1,1,8) (1,27,1) (1,27,8) (27,1,1) (27,1,8) (27,27,1) (27,27,8).

3. Subroutine MED Fortran

This subroutine contains the semi-discrete form of the explicit finite difference equation for the medium grid. The parameters for this grid are listed in Table 4. The subroutine is broken into thirteen sections, each dealing with a different type of node. These nodes are shown in Figure 33 and Figure 34, where the letters correspond to the following sections. The constants in Table 4 are as normally defined for Fourier number and Biot number. Note that the Biot number is with respect to Δz, not Δx and that Δz is one half of Δx for a surface node.

The medium grid subroutine is the most complex because it must handle interfacing with both the fine and the coarse grids. The distance to a coarse node is twice that of normal medium node spacing. Because the change in temperature at a node is related to $1/\Delta x$, this reduces the effect by one half. With interfacing to the fine grid, the spacing is two-thirds of normal medium node spacing. In this case the effect is increased by three-halves. Interfacing with the fine grid also requires one additional step. Since one medium node is in contact with nine fine grid nodes (six for surface nodes), it is necessary to take the average temperature of the fine nodes. The second difficulty in interfacing is keeping track of the medium nodal points location with respect to the coarse and fine grids. In the subroutine all coarse array indexes are combined with an A to indicate they are A array subscripts and all fine array indexes are combined with a C to indicate they are C array subscripts. It is noted that the coarse, A, array is only two dimensional.

a. The Interior Nodes

The equation for the interior node is the simplest. This equation is valid over the following range: i = 2 to 26, j = 2 to 26 and k = 2 to 8 with the exception any node adjacent to the fine zone, which must be handled separately. The zone formed by the nodes excluded is complicated and is as follows: for $k \leq 4$ then i = 10 to 18 and

Table 4. PARAMETERS DEFINING THE MEDIUM GRID

Dimensions: 81mm x 81mm x 27mm Node Volume: 3mm x 3mm x 3mm Array: B(27,27,10)
The following constants are used: $FO(2) = 1.4722\,\Delta t$ $BI(2) = 1.132x10^{-3}$ $K = 53.0\ W/mK$ $Cp = 4.0x10^{6}\ J/m^{3}K$ $h = 10.0\ W/m^{2}K$

j = NC to NC+8, also for k < 4 with i = 9 or 19 and j = NC to NC+8, and also for k <4 with i= 10 to 18 and j = NC-1 or NC+9.

b. The Top and Bottom Surface Nodes

The surface nodes have the +k or -k index term replaced by a surface convection term. In addition, the top surface has an exclusion zone caused by the fine grid. The equations apply over the following range: i = 2 to 26, j = 2 to 26 and k = 1 or 10. At the surface, k = 1, the exclusion zone is similar to the interior nodes: i = 10 to 18 and j = NC to NC+8 also with i = 9 or 19 then j = NC to NC+8, and with i= 10 to 18 then j = NC-1 or NC+9.

c. The Exterior End Faces (Coarse Interface)

The end face equations have either a +i or -i index term replaced with a difference term from the coarse zone. Due to the further distance of a coarse nodal point it is multiplied by one-half. These equations are valid over the following range: i = 2 to 26, j = 1 or 27, k = 2 to 9, IA = 7 to 15 and JA = NB-1 or NB+9.

d. The Exterior Side Faces (Coarse Interface)

The side face equations have either a +j or -j index term replaced with a difference term from the coarse zone. Due to the further distance of a coarse nodal point it is multiplied by one-half. These equations are valid over the following range: i = 1 or 27, j = 2 to 26, k = 2 to 9, IA = 6 or 16 and JA = NB to NB+8.

e. The Exterior End Edges (Coarse Interface)

The end edge equations have either a +i or -i index term replaced with a difference term from the coarse zone and a +k or -k index term replaced by a surface

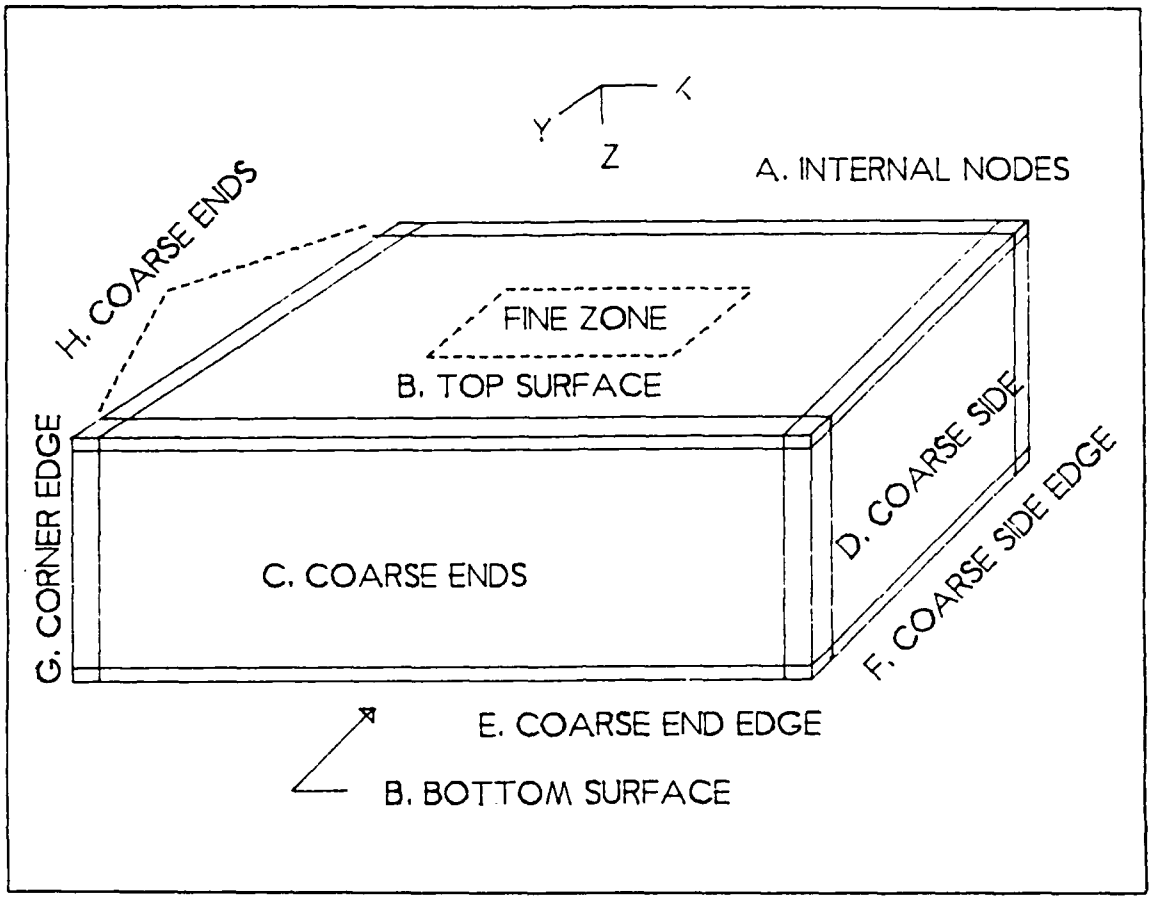


Figure 33. The Medium Grid Node Types (Coarse Interface): The letters correspond the sections of the subroutine MED.

convection term. These equations are valid over the following range: i = 2 to 26, j = 1 or 27, k = 1 or 10, IA = 7 to 15 and JA = NB-1 or NB+9.

f. The Exterior Side Edges (Coarse Interface)

The side edge equations have either a +j or -j index term replaced with a difference term from the coarse zone and a +k or -k index term replaced by a surface convection term. These equations are valid over the following range: i = 1 or 27, j = 2 to 26, k = 1 or 10, IA = 6 or 16 and JA = NB to NB+8.

g. The Exterior Corner Edges (Coarse Interface)

The corner edges have both a +i or -i index and a +j or -j index terms replaced with a difference term from the coarse zone. These equations are valid over the

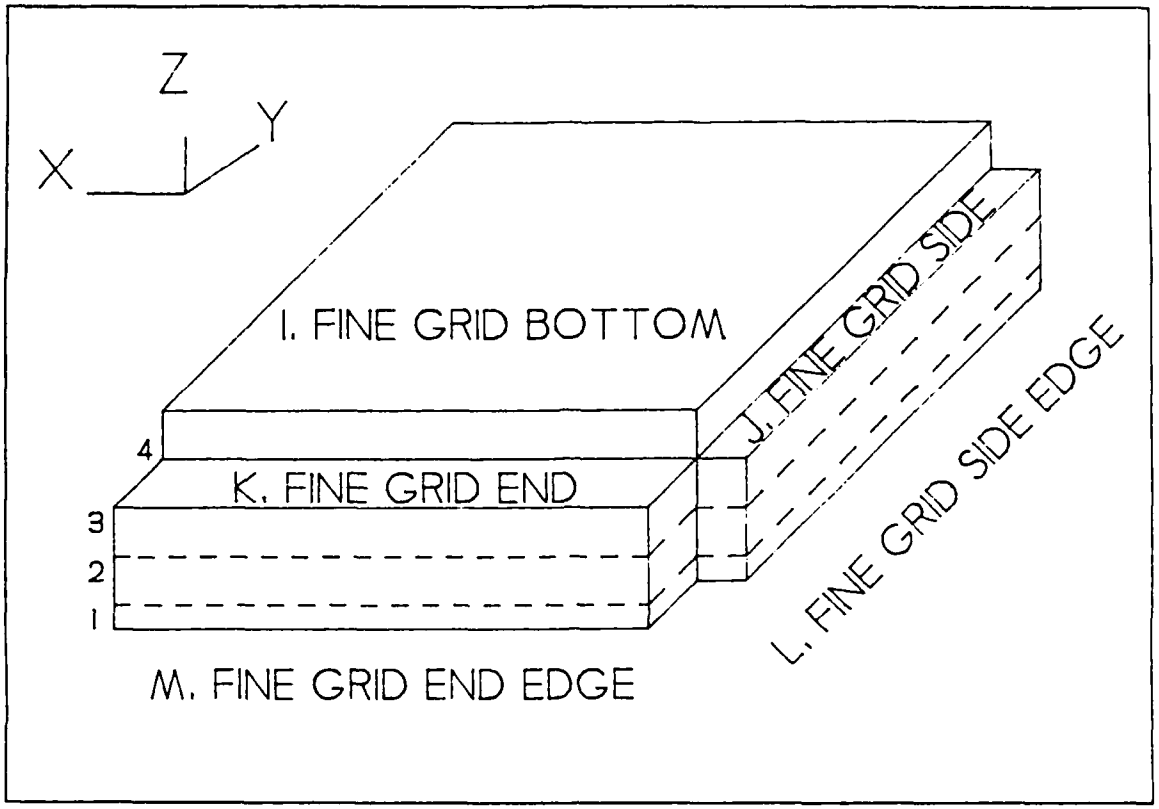


Figure 34. The Medium Grid Node Type (Fine Interface): The figure is orientated upside-down to show the medium nodes which are adjacent to the fine zone. The letters correspond the sections of the subroutine MED.

following range: i = 1 or 27, j = 1 or 27, k = 2 to 9 and both 1) IA = 7 or 15 and JA = NB-1 or NB+9 and 2) IA = 6 or 16 and JA = NB or NB+8.

h. The Exterior Corners (Coarse Interface)

The eight corners are all distinct equations. They each have two terms replaced by a difference term from the coarse zone and one term replaced by a surface convection term. The equations are for the following nodal points: (1,1,1) (1,1,10) (1,27,1) (1,27,10) (27,1,1) (27,1,10) (27,27,1) (27,27,10).

i. The Bottom of the Exclusion Zone (Fine Interface)

For this equation the -k index term is replace with a difference term with nine fine zone nodes. This equation is valid for the following region: i = 10 to 18, j = NC to NC+8, k = 4, IC = 1 to 27, JC = 1 to 27 and KC = 8.

j. The Sides of the Exclusion Zone (Fine Interface)

For this equation the +i or -i index term is replaced with a difference term with nine fine zone nodes. These equations are for the following region: i = 9 or 19, j = NC to NC+8, k = 2 to 3, IC = 1 or 27, JC = 1 to KC = 3 to 8.

k. The Ends of the Exclusion Zone (Fine Interface)

For this equation the +j or -j index term is replaced with a difference term with nine fine zone nodes. These equations are for the following region: i = 10 to 18, j = NC-1 or NC+9, k = 2 to 3, IC = 1 to 27, JC = 1 or 27 and KC = 3 to 8.

l. The Side Edge of the Exclusion Zone (Fine Interface)

For this equation the +i or -i index term is replaced with a difference term with six fine nodes and a -k index term is replaced with a surface convection term. These equations are for the following 27 and region: i = 9 or 19, j = NC to NC+8, k = 1, IC = 1 or 27, JC = 1 to 27 and KC = 1 to 2.

m. The End Edge of the Exclusion Zone (Fine Interface)

For this equation the +j or -j index term is replaced with a difference term with six fine nodes and a -k index term is replaced with a surface convection term. These equations are for the following region: i = 10 to 18, j = NC-1 to NC+9, k = 1, IC = 1 to 27, JC = 1 or 27 and KC = 1 to 2.

4. Subroutine COR Fortran

This subroutine contains the semi-discrete form of the explicit finite difference equation for the coarse grid. The parameters for this grid are listed in Table 5. The subroutine is broken into six sections, each dealing with a different type of node. These nodes are shown in Figure 35, where the letters correspond to the following sections. The constants in Table 5 are as normally defined for Fourier number and Biot number. Note the area of a node control volume surface is one-third the area of its side and that each node control volume has two surfaces. This modifies the Biot number to two-thirds of what it would be using a Δx spacing. Since the grid is two dimensional, all nodes have a surface convection term.

The coarse grid subroutine is the simplest since it is two dimensional. On interfacing to the medium grid, the medium grid at the interface is two-thirds the distance of a normal coarse node. This increase the effect three-halves. Interfacing with the medium grid also requires one additional step. Since one coarse node is in contact with 30 medium grid nodes it is necessary to take the average of the medium nodes. The second difficulty in interfacing is keeping track of the medium nodal points location with

Table 5. PARAMETERS DEFINING THE COARSE GRID

Dimensions: 180mm x 315mm x 27mm Node Volume: 9mm x 9mm x 27mm Array: A(21,36)
The following constants are used: $FO(1) = 0.1635\,\Delta t$ $BI(1) = 1.132x10^{-3}$ $K = 53.0\ W/mK$ $Cp = 4.0x10^{6}\ J/m^{3}K$ $h = 10.0\ W/m^{2}K$

respect to the coarse grid. In the subroutine all medium array indexes are combined with a B to indicate they are B array subscripts.

a. The Corners

The selection criterion of the plate size was that the edges of the plate would not have a significant temperature change. Thus the four corners are assumed clamped and hence adiabatic. The indexes for the four corner are: (1,1) (1,36) (21,1) (21,36).

b. The Ends

It is assumed that negligible heat is lost out of the ends and hence either the +j or -j index term is dropped. These equations are valid over the following range: i = 2 to 20, j = 1 or 36.

c. The Sides

It is assumed that negligible heat is lost out of the sides and hence either the +i or -j index term is dropped. These equations are valid over the following range: i = 1 or 21, j = 2 to 35.

d. The Interior Nodes

This is the basic equation and is valid over the following range: i = 2 to 20, j = 2 to 35 with the exception of any node adjacent to the medium zone. The zone formed by the nodes excluded is as follows: i = 7 to 15 and j = NB to NB+8, also with i = 6 or 16 and j = NB to NB+8, and also with i= 7 to 15 and j = NB-1 or NB+9.

e. The Ends of the Exclusion Zone (Medium Interface)

For this equation the +j or -j index term is replaced with a difference term with the 30 medium nodes. Care is taken to note that the surface medium nodes have

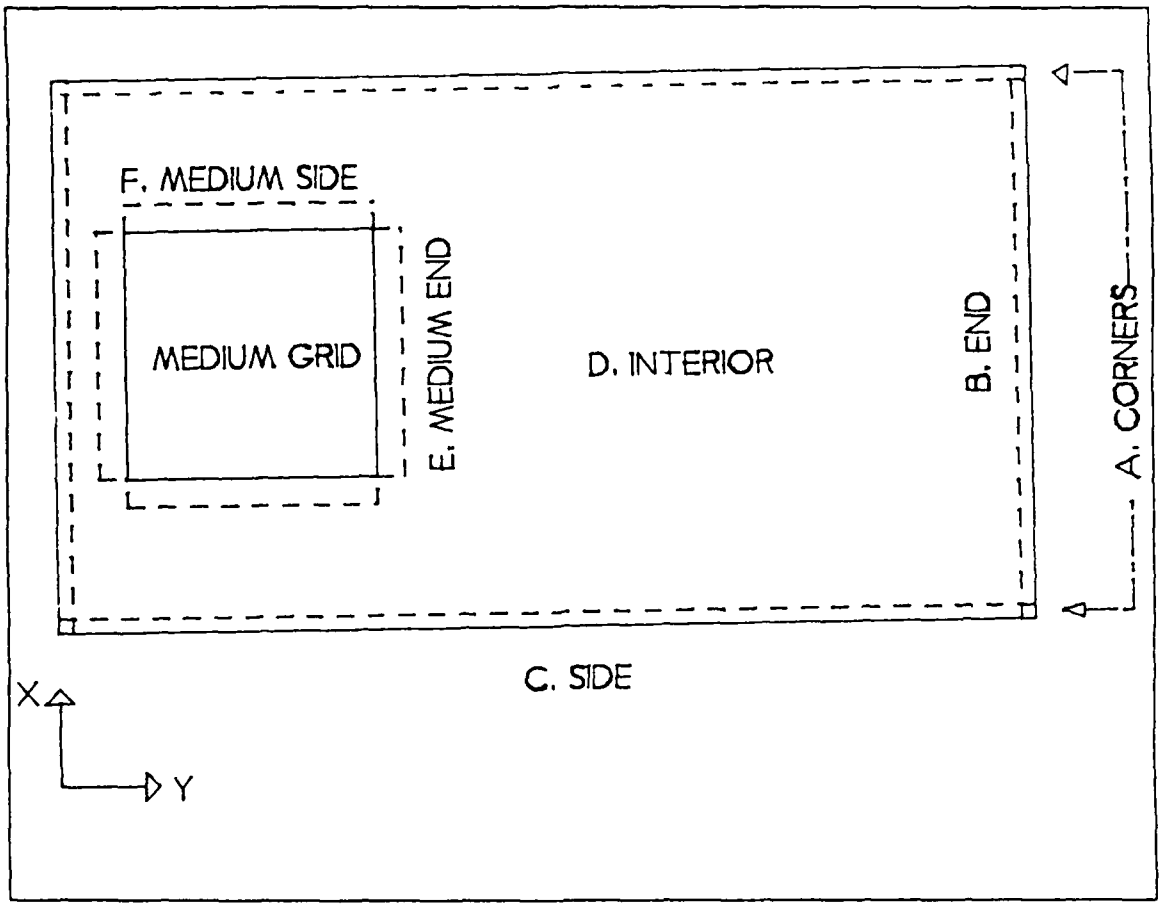


Figure 35. The Coarse Grid Node Types: The letters correspond to the subroutine COR.

only have the volume of a regular medium node. These equations are valid over the range: i = 7 to 15, j = NB-1 or NB+9, IB = 1 to 27, JB = 1 or 27 and KB = 1 to 10.

f. The Sides of the Exclusion Zone (Medium Interface)

For this equation the +i or -i index term is replaced with a difference term with the 30 medium nodes. Care is taken to note that the surface medium nodes have only have the volume of a regular medium node. These equations are valid over the range: i = 6 or 16, j = NB to NB+8, IB = 1 or 27, JB = 1 to 27 and KB = 1 to 10.

B. USER'S GUIDE TO MODIFYING AND RUNNING THE WELD PROGRAM

The program WELD and its variants discussed in Appendix C are not interactive programs. Due to the amount of machine time that a typical simulation required, these

programs were run on the grave shift by the batch processor, VMSCHED. Prior to each run, the program starting parameters were reviewed, and the program was recompiled. In Table 2 the major program variables are listed. There are several major changes that the user can decide to make before running the program:

- The time for the program to run
- Whether the run is new or a restart of a previous run
- The arc velocity in the y and x directions
- The arc power level
- The starting temperatures
- The material properties
- The arc shape
- Programming the arc movement and power history
- Changing the grid sizes

The first four of these are very simple to perform and are usually checked prior to each run. The last five are more complicated, and some guidance will be given about how to successfully change these parameters.

1. Setting up for a normal run

This section discusses how to set the time to run the program, the type of run, the arc power and speed. The time to run is specified by the variable FINI, just set this variable equal to the amount of time to simulate, in seconds. The next choice is to start a new simulation or restart an old one. To start a new simulation, insure that no old output files are on your A disk, and comment out the statement 'GOTO 100'. To restart a old problem, insure that the output files are on your A disk and make the statement 'GOTO 100' active. For problems which are not a restart you can also check the following. The arc velocity is set by the variable YVEL. This is the speed of the arc in millimeters per second. In the WELDMA version of the program, you can also set an x velocity, XVEL. However in the program WELD this variable is not used. The arc power level is the product of three variables; VOLT, AMP, and EFF. Where VOLT is in volts, AMP is in amperes, and EFF is a fraction of one, for the arc efficiency. This gives the power in watts.

2. Setting the starting temperatures

There are three places the starting temperatures are set. The first two are in the DATA statement. The A and B arrays (coarse and medium grids) are set by specifying

a temperature in degrees Kelvin. For the program listing below they are set at 300 degrees Kelvin. The starting temperature for the C array (fine grid) is specified by its enthalpy in watts per meter cubed. For the plain carbon steel being modeled, this is $1.14237x10^8\ W/m^3$ at 300 degrees Kelvin. The last temperature that can be set is the temperature of the surroundings by the variable TINF. This also is in degree Kelvin.

3. Changing the material properties

In Table 3, Table 4, and Table 5, care was made to always define constants used for the Fourier and Biot numbers in each grid. In the program there is one block in which these numbers are defined. By changing these constants, FO and BI, and setting the interface thermal conductivity, FKB, for the FIN subroutine, the material properties and the surface conditions are changed. For the fine zone, which uses variable thermal properties, it is necessary to reprogram the three functions FK(T), T(H) and H(T). At present these are piece-wise continuous linear functions, but any function can be used. It is recommended that they be kept simple since they are repeatedly called by the program.

When the material properties or mesh spacing are changed, the Fourier number is also changed. This may require changing the step size to insure the stability requirements are met, as discussed in Appendix B. When DELT is changed, it will also require changing the calculation of NDIV. These will be required to be changed not only at the start of the program, but for restarts, in the restart section as well. If you will be changing DELT often, you may desire to change the program to calcuate NDIV as a function of DELT directly.

4. Shaping the arc power distribution

The program is written to allow shaping the arc as a double-elliptical gaussian distribution. This feature usually used a spherical gaussian distribution because arc shaping was not found to be necessary. Additionally it was feared that arc shaping would interfere with studying transient problems, since this would superimpose this empirical steady-state solution. The general form of the equation is given by equation A.2, where the coefficients a, b, and c are the desired radial dimensions of the arc in the x, y, and z directions, respectively. The distances are in millimeters. In the source code, the use of different radii in the y direction has been commented out. This shows how the double elliptical pattern could be inputted for the arc power distribution. Note that it is required to change the source code twice. Once for calculating the volume averaging factor, SUM. And a second time for where the power is actually added to the nodes in the model.

5. Programming the arc movement and power history

The arc movement is presently determined by two variables. YARC and DIS. Both of these variables are functions of YVEL*TIME. This product can be replace by any desired function in both YARC and DIS defining equations. Thus arc accelerations, starts and stops can be programmed in as desired. It is important to change both of these variables to keep track of the actual distance the arc has traveled.

The arc power is in general constant and is inputted by the variable Q, which is the change of enthalpy of a standard node at one time step, that is in Joules per meter cubed. The easiest way to change Q is probably by multiplying it by a gain factor in the main DO loop. Thus, as the program runs you can change the arc power as desired. If the size of the grid spacing is being changed (i.e. from one millimeter spacing to 2 millimeter spacing) then the variable Q would also be effected. For the example of two millimeters, then Q = QDOT*DELT*1.25E8.

6. *Changing the grid geometry*

To change the number of nodes is the most difficult change to perform. To change just the spacing of the grid, without changing the number of nodes, is just equivalent to changing the material properties. Just recalculate all of the material coefficients, FO and BI. In addition you will need to change the dimensions used in adding power; both for calculating the gaussian distribution as well as the change in enthalpy variable, Q.

If the number and distribution of nodes is to be changed, than a major rewrite of the program is required. This would require checking every equation in the three subroutines FIN, MED and COR as well as checking the main program sections which handle the input, output, Runge-Kutta sum, and grid shifting. An example of where this has been done is included in Appendix C. This is the version WELDC, where the fine and medium grids have been elongated to allow studing cooling rates. The modification approximately doubled the number of nodal points and hence doubled the run time of the program.

C. SOURCE LISTING OF PROGRAM WELD FORTRAN

```
      PROGRAM WELD
**********************************************************************
*                                                                    *
* DATE:   16 AUGUST 1988      WRITTEN BY:   ROBERT ULE               *
*                                                                    *
**********************************************************************
      DIMENSION A(21,36),AOUT(21,36),AIN(21,36),ASUM(21,36),
     *B(27,27,10),BOUT(27,27,10),BIN(27,27,10),BSUM(27,27,10),
```

```
     *C(27,27,8),COUT(27,27,8),CIN(27,27,8),CSUM(27,27,8),
     *FO(3),BI(4)
      INTEGER NDIV,I,J,K,M,BSTEP,CSTEP
      CHARACTER ANS
      LOGICAL YES
      COMMON NB,NC,YARC,SUM,TINF,BI,FO

      DATA A,B/8046*300.0/,C/5832*1.14237E8/,ASUM,BSUM/8046*0.0/
     *,AOUT,BOUT/8046*0.0/,CSUM/5832*0./,COUT/5832*0./

C  SET TIME TO RUN THE PROBLEM AND FIXED CONDITIONS

      FINI = 5.00
      NDIV=FINI*100
      XVEL=.0
      DELT=.01
      TINF=300.0

C  IF THIS IS A RESTART OF A PREVIOUS PROBLEM GOTO 100

C     GOTO 100

C  SET INITIAL VALUES FOR STARTING A PROBLEM

      YVEL=4.
      VOLT=30.
      AMP=265.
      EFF=.32

C  OPEN THE OUTPUT FILES

      OPEN(1,FILE='SURF',STATUS='NEW',FORM='UNFORMATTED')
      OPEN(2,FILE='FINAL',STATUS='NEW',FORM='UNFORMATTED')
      OPEN(3,FILE='CUT',STATUS='NEW',FORM='UNFORMATTED')

C  THE INITIAL CONDITIONS

      TIME=0.
      STEP=3.
      OUT=.49
      N=0
      BSTEP=0
      CSTEP=0
      NB=3
      NC=10

C  WELD PARAMETERS

      QDOT=EFF*VOLT*AMP
      Q=QDOT*DELT*1.E9

C  REENTRY FOR RESTART

  200 CONTINUE

C  BOUNDARY CONDITION COEFFICIENTS
```

```
      FO(1)=DELT*.1636
      FO(2)=DELT*1.4722
      FO(3)=DELT*1000000.
      BI(1)=.001132
      BI(2)=.001132
      BI(3)=DELT*.00009299
      BI(4)=DELT*50000.

C  CALCULATE THE RUNGE-KUTTA APPROXIMATION

      DO 10 M=1,NDIV
       TIME=TIME+DELT
       DIS=TIME*YVEL
       N=N+1

C  POSITION HEAT SOURCE AND CALCULATE VOLUME WEIGHTING FACTOR   SUM'

       YARC=YVEL*TIME+73-NB*9-NC*3
       LOF=65-INT(VEL*TIME)
       SUM=0.
       DO 1 J=7,23
C       IF ((J-YARC).GT.(0.0)) THEN  **** Allowed shaping the arc.
         SUM=SUM+QDENSE/4.*EXP(-.10625*((J-YARC)**2))
C       ELSE
C         SUM=SUM+QDENSE/10.*EXP(-.017*((J-YARC)**2))
C       ENDIF
    1  CONTINUE
       SUM=SUM/Q

C  ADD THE HEAT FROM THE ARC

      DO 2 J=7,23
C       IF ((J-YARC).GT.(0.0)) THEN  **** Allowed shaping the arc
         Y=.25*EXP(-.10625*((J-YARC)**2))
C       ELSE
C         Y=.1*EXP(-.017*((J-YARC)**2))
C       ENDIF
        DO 2 I=11,17
         DO 2 K=1,3
          C(I,J,K)=C(I,J,K)+Y*XZ(I-10,K)/SUM
    2 CONTINUE

C  CONVERT ENTHALPY TO TEMPERATURE FOR THE FINE PLATE

      DO 60 I=1,27
       DO 60 J=1,27
        DO 60 K=1,8
         CIN(I,J,K)=T(C(I,J,K))
   60 CONTINUE

C  FIRST RUNGE-KUTTA ITERATION

      CALL COR(A,B,ASUM)
      CALL MED(A,B,CIN,BSUM)
      CALL FIN(B,CIN,CSUM)
```

```
C  FIND T+K1/2

      DO 11 I=1,21
       DO 11 J=1,36
11    AIN(I,J)=A(I,J)+ASUM(I,J)/2.
      DO 12 I=1,27
       DO 12 J=1,27
        DO 13 K=1,10
13      BIN(I,J,K)=B(I,J,K)+BSUM(I,J,K)/2.
        DO 12 K=1,8
        CIN(I,J,K)=T(C(I,J,K)+CSUM(I,J,K)/2.)
12 CONTINUE

C  SECOND ITERATION

      CALL COR(AIN,BIN,AOUT)
      CALL MED(AIN,BIN,CIN,BOUT)
      CALL FIN(BIN,CIN,COUT)

C  FIND T+K2/2

      DO 21 I=1,21
       DO 21 J=1,36
       ASUM(I,J)=ASUM(I,J)+2.*AOUT(I,J)
21    AIN(I,J)=A(I,J)+AOUT(I,J)/2.
      DO 22 I=1,27
       DO 22 J=1,27
        DO 23 K=1,10
        BSUM(I,J,K)=BSUM(I,J,K)+2.*BOUT(I,J,K)
23      BIN(I,J,K)=B(I,J,K)+BOUT(I,J,K)/2.
        DO 22 K=1,8
        CSUM(I,J,K)=CSUM(I,J,K)+2.*COUT(I,J,K)
        CIN(I,J,K)=T(C(I,J,K)+COUT(I,J,K)/2.)
22 CONTINUE

C  THIRD ITERATION

      CALL COR(AIN,BIN,AOUT)
      CALL MED(AIN,BIN,CIN,BOUT)
      CALL FIN(BIN,CIN,COUT)

C  FIND T+K3

      DO 31 I=1,21
       DO 31 J=1,36
       ASUM(I,J)=ASUM(I,J)+2.*AOUT(I,J)
31    AIN(I,J)=A(I,J)+AOUT(I,J)
      DO 32 I=1,27
       DO 32 J=1,27
        DO 33 K=1,10
        BSUM(I,J,K)=BSUM(I,J,K)+2.*BOUT(I,J,K)
33      BIN(I,J,K)=B(I,J,K)+BOUT(I,J,K)
        DO 32 K=1,8
        CSUM(I,J,K)=CSUM(I,J,K)+2.*COUT(I,J,K)
        CIN(I,J,K)=T(C(I,J,K)+COUT(I,J,K))
```

```
   32 CONTINUE

C  FORTH ITERATION

      CALL COR(AIN,BIN,AOUT)
      CALL MED(AIN,BIN,CIN,BOUT)
      CALL FIN(BIN,CIN,COUT)

C  PERFORM THE RUNGE-KUTTA SUM

      DO 40 I=1,21
       DO 40 J=1,36
   40  A(I,J)=A(I,J)+(ASUM(I,J)+AOUT(I,J))/6
      DO 41 I=1,27
       DO 41 J=1,27
        DO 43 K=1,10
   43   B(I,J,K)=B(I,J,K)+(BSUM(I,J,K)+BOUT(I,J,K))/6
        DO 41 K=1,8
   41   C(I,J,K)=C(I,J,K)+(CSUM(I,J,K)+COUT(I,J,K))/6

C  STEP GRID IF REQUIRED

      IF (DIS.GE.STEP) THEN
       STEP=STEP+3.
        BSTEP=BSTEP+1

C  MOVE FINE GRID, FIRST AVERAGE FINE TEMPS AND ASSIGN TO MED GRID

        DO 51 IB=1,9
         TB1=0.
         TB2=0.
         TB3=0.
         DO 50 IC=IB*3-2,IB*3
          DO 50 JC=1,3
           TB1=T(C(IC,JC,1))+2.*T(C(IC,JC,2))+TB1
           DO 50 KC=3,5
            TB2=TB2+T(C(IC,JC,KC))
            TB3=TB3+T(C(IC,JC,KC+3))
   50     CONTINUE
          B(IB+9,NC,1)=TB1/27
          B(IB+9,NC,2)=TB2/27
          B(IB+9,NC,3)=TB3/27
   51   CONTINUE

C  SHIFT FINE GRID

        DO 52 J=1,24
         JJ=J+3
         DO 52 I=1,27
          DO 52 K=1,8
   52     C(I,J,K)=C(I,JJ,K)

C  ASSIGN MED TEMPS TO FINE GRID

        DO 53 I=10,18
         TB1=B(I,NC+9,1)
```

```
            TB2=B(I,NC+9,2)
            TB3=B(I,NC+9,3)
            DO 53 IC=(I-9)*3-2,(I-9)*3
             DO 53 JC=25,27
              C(IC,JC,1)=H(TB1)
              C(IC,JC,2)=H(TB1)
              DO 53 KC=3,5
               C(IC,JC,KC)=H(TB2)
               C(IC,JC,KC+3)=H(TB3)
   53     CONTINUE
          NC=NC+1

C  CHECK TO SEE IF TIME TO SHIFT MEDIUM GRID
          IF (BSTEP.EQ.3) THEN
           BSTEP=0

C  THEN AVERAGE MEDIUM GRID AND ASSIGN TO COARSE GRID

            DO 54 IA=1,9
             TA1=0.
             DO 55 IB=IA*3-2,IA*3
              DO 55 JB=1,3
               TA1=TA1+(B(IB,JB,1)+B(IB,JB,10))/2
               DO 55 KB=2,9
                TA1=TA1+B(IB,JB,KB)
   55        CONTINUE
             A(IA+6,NB)=TA1/81
   54       CONTINUE

C  SHIFT MED GRID

            DO 56 J=1,24
             JJ=J+3
             DO 56 I=1,27
              DO 56 K=1,10
   56         B(I,J,K)=B(I,JJ,K)

C  ASSIGN COARSE TEMPS TO MEDIUM GRID

            DO 57 I=7,15
             TA1=A(I,NB+9)
             DO 57 IB=(I-6)*3-2,(I-6)*3
              DO 57 JB=25,27
               DO 57 KB=1,10
                B(IB,JB,KB)=TA1
   57       CONTINUE
            NB=NB+1
            NC=10
           ENDIF
         ENDIF

C  OUTPUT SOLUTION FILES

         IF (TIME.GE.OUT) THEN
          OUT=OUT+.5
```

```
      WRITE(1) ((T(C(I,J,1)),I=1,27),J=1,27)
C  AVERAGE FINE TEMPS AND ASSIGN TO MEDIUM GRID
      DO 61 J=0,8
       DO 61 IB=1,9
        TB1=0.
        DO 66 IC=IB*3-2,IB*3
         DO 66 JC=J*3+1,J*3+3
66       TB1=T(C(IC,JC,1))+2*T(C(IC,JC,2))+TB1
61      B(9+IB,NC+J,1)=TB1/27.
      WRITE(1)((B(I,J,1),I=1,27),J=1,27)
      WRITE(3)((T(C(I,INT(YARC),K)),I=1,27),K=1,8)
     ENDIF
     IF (NB.LE.11) THEN
      IB=26-(NB-3)*3
      IF (IB.GE.NC.AND.IB.LE.NC+8) THEN
       IC=2+(IB-NC)*3
       WRITE(4) (T(C(IC,J,1)),J=2,26,3)
      ELSE
       WRITE(4) (B(IB,J,1),J=10,18)
      ENDIF
     ENDIF

10 CONTINUE

C  OUTPUT FINAL RESULTS

     WRITE(2) TIME
     WRITE(2)(((T(C(I,J,K)),I=1,27),J=1,27),K=1,8)
C  AVERAGE FINE TEMPERATURES AND ASSIGN TO MEDIUM GRID
     DO 62 J=0,8
      DO 62 IB=1,9
       TB1=0.
       TB2=0.
       TB3=0.
       DO 63 IC=IB*3-2,IB*3
        DO 63 JC=1+J*3,3+J*3
         TB1=T(C(IC,JC,1))+2.*T(C(IC,JC,2))+TB1
         DO 63 KC=3,5
          TB2=TB2+T(C(IC,JC,KC))
          TB3=TB3+T(C(IC,JC,KC+3))
63     CONTINUE
       B(IB+9,NC+J,1)=TB1/27.
       B(IB+9,NC+J,2)=TB2/27.
       B(IB+9,NC+J,3)=TB3/27.
62 CONTINUE
     WRITE(2)(((B(I,J,K),I=1,27),J=1,27),K=1,10)
C  AVERAGE MEDIUM TEMPERATURES AND ASSIGN TO COARSE GRID
     DO 64 J=0,8
      DO 64 IA=1,9
       TA1=0.
       DO 65 IB=IA*3-2,IA*3
        DO 65 JB=1+J*3,3+J*3
         TA1=TA1+(B(IB,JB,1)+B(IB,JB,10))/2
         DO 65 KB=2,9
          TA1=TA1+B(IB,JB,KB)
65   CONTINUE
```

```
          A(IA+6,NB+J)=TA1/81
       64 CONTINUE
          WRITE(2)((A(I,J),I=1,21),J=1,36)

C  ALLOW OPTION OF RESTARTING THE PROBLEM

          INQUIRE(9,OPENED=YES)
          IF(YES) THEN
           REWIND(9)
          ELSE
           OPEN(9,FILE='RESTAR',STATUS='NEW',FORM='UNFORMATTED')
          ENDIF
          WRITE(9)OUT,Q,NB,NC,YVEL,STEP,CSTEP,BSTEP,N,DELT
          WRITE(9)A
          WRITE(9)B
          WRITE(9)C
          STOP

C  PROCEDURE FOR RESTARTING A PREVIOUS PROBLEM

      100 OPEN(2,FILE='FINAL',STATUS='OLD',FORM='UNFORMATTED')

C  THE INITIAL CONDITIONS

          READ(2)TIME
          REWIND(2)
          IF (TIME.LT.(.49)) THEN
           OPEN(1,FILE='SURF',STATUS='NEW',FORM='UNFORMATTED')
           OPEN(3,FILE='CUT',STATUS='NEW',FORM='UNFORMATTED')
          ELSE
           OPEN(1,FILE='SURF',STATUS='OLD',FORM='UNFORMATTED')
           OPEN(3,FILE='CUT',STATUS='OLD',FORM='UNFORMATTED')
          ENDIF
          OPEN(9,FILE='RESTAR',STATUS='OLD',FORM='UNFORMATTED')
          READ(9)OUT,Q,NB,NC,YVEL,STEP,CSTEP,BSTEP,N,DELT

C  POSITION THE RUNNING OUTPUT FILES TO THE END OF THE FILES

          DO 102 L=1,N/50
           READ(1)((C(I,J,1),I=1,27),J=1,27)
           READ(1)((B(I,J,1),I=1,27),J=1,27)
           READ(3)((C(I,1,K),I=1,27),K=1,8)
      102 CONTINUE

C  INPUT THE TEMPERATURE/ENTHALPY MATRICES

          READ(9)A
          READ(9)B
          READ(9)C
          GOTO 200
          END

C   FUNCTION FOR K AS A VARIABLE OF TEMPERATURE

          REAL FUNCTION FK(T)
          IF (T.LT.973.) THEN
```

```
      FK=63.92-.04*T
     ELSE
      IF (T.LT.1773.) THEN
       FK=18.91875+.00625*T
      ELSE
       FK=90.
      ENDIF
     ENDIF
     RETURN
     END

C  FUNCTION FOR FINDING TEMPERATURE AS A FUNCTION OF ENTHALPY

     REAL FUNCTION T(H)
     IF (H.LT.2.75E9) THEN
      T=2.3636E-7*H+273
     ELSE
      IF (H.LT.3.25E9) THEN
       T=5.E-8*H+785.5
      ELSE
       IF (H.LT.7.5E9) THEN
        T=1.8235E-7*H+355.35
       ELSE
        IF (H.LT.10.25E9) THEN
         T=1.8182E-8*H+1586.64
        ELSE
         T=2.222E-7*H-504.45
        ENDIF
       ENDIF
      ENDIF
     ENDIF
     RETURN
     END

C  FUNCTION FOR FINDING ENTHALPY AS A FUNCTION OF TEMPERATURE

     REAL FUNCTION H(T)
     IF (T.LT.923.) THEN
      H=4.231E6*T-1.15506E9
     ELSE
      IF (T.LT.948.) THEN
       H=.02E9*T-15.71E9
      ELSE
       IF (T.LT.1723.) THEN
        H=5.484E6*T-1.9488E9
       ELSE
        IF (T.LT.1773.) THEN
         H=.055E9*T-87.265E9
        ELSE
         H=.0045E9*T+2.2715E9
        ENDIF
       ENDIF
      ENDIF
     ENDIF
     END
```

D. SOURCE LISTING OF SUBROUTINE FIN FORTRAN

```
      SUBROUTINE FIN(B,C,COUT)

      DIMENSION B(27,27,10),C(27,27,8),COUT(27,27,8),FO(3),BI(4)
      COMMON NB,NC,YARC,SUM,TINF,BI,FO

      GK(T1,T2)=FK(T1)*(T1-T2)/(FK(T1)+FK(T2))

      DATA FKB/26.5/

*  A. THE INTERIOR NODES

      DO 1 I=2,26
       DO 1 J=2,26
        DO 1 K=2,7
         COUT(I,J,K)=FK(C(I,J,K))*FO(3)*(GK(C(I+1,J,K),C(I,J,K))+
     *   GK(C(I-1,J,K),C(I,J,K))+GK(C(I,J+1,K),C(I,J,K))+GK(C(I,J-1,K),
     *   C(I,J,K))+GK(C(I,J,K+1),C(I,J,K))+GK(C(I,J,K-1),C(I,J,K)))
    1 CONTINUE

*  B. THE TOP SURFACE NODES

      TINF4=TINF**4
      DO 2 I=2,26
       DO 2 J=2,26
        COUT(I,J,1)=FK(C(I,J,1))*FO(3)*(GK(C(I+1,J,1),C(I,J,1))+
     *   GK(C(I-1,J,1),C(I,J,1))+GK(C(I,J+1,1),C(I,J,1))+
     *   GK(C(I,J-1,1),C(I,J,1))+2.*GK(C(I,J,2),C(I,J,1)))+
     *   BI(3)*(TINF4-C(I,J,1)**4)+BI(4)*(TINF-C(I,J,1))
    2 CONTINUE

*  C. THE BOTTOM SURFACE NODES (MEDIUM INTERFACE)

      DO 3 I=2,26
       IB=(I-.5)/3+10
       DO 3 J=2,26
        JB=(J-.5)/3+NC
        COUT(I,J,8)=FO(3)*(FK(C(I,J,8))*(GK(C(I+1,J,8),C(I,J,8))+
     *   GK(C(I-1,J,8),C(I,J,8))+GK(C(I,J+1,8),C(I,J,8))+GK(C(I,J-1,8),
     *   C(I,J,8))+GK(C(I,J,7),C(I,J,8)))+FKB*(B(IB,JB,4)-C(I,J,8)))
    3 CONTINUE

*  D. THE END FACES (MEDIUM INTERFACE)

      JB=NC-1
      JMAX=NC+9
      DO 4 I=2,27
       IB=(I-.5)/3+10
       DO 4 K=2,7
        KB=2
        IF(K.EQ.2) KB=1
        IF(K.GE.6) KB=3
        COUT(I,1,K)=FO(3)*(FK(C(I,1,K))*(GK(C(I+1,1,K),C(I,1,K))+
     *   GK(C(I-1,1,K),C(I,1,K))+GK(C(I,2,K),C(I,1,K))+GK(C(I,1,K+1),
     *   C(I,1,K))+GK(C(I,1,K-1),C(I,1,K)))+FKB*(B(IB,JB,KB)-
```

```
*    C(I,1,K)))
     COUT(I,27,K)=FO(3)*(FK(C(I,27,K))*(GK(C(I+1,27,K),C(I,27,K))+
*    GK(C(I-1,27,K),C(I,27,K))+GK(C(I,26,K),C(I,27,K))+
*    GK(C(I,27,K+1),C(I,27,K))+GK(C(I,27,K-1),C(I,27,K)))+
*    FKB*(B(IB,JMAX,KB)-C(I,27,K)))
   4 CONTINUE

*  E. THE SIDE FACES (MEDIUM INTERFACE)

      DO 5 J=2,26
       JB=(I-.5)/3+NC
       DO 5 K=2,7
        KB=2
        IF(K.EQ.2) KB=1
        IF(K.GE.6) KB=3
        COUT(1,J,K)=FO(3)*(FK(C(1,J,K))*(GK(C(2,J,K),C(1,J,K))+
*    GK(C(1,J+1,K),C(1,J,K))+GK(C(1,J-1,K),C(1,J,K))+
*    GK(C(1,J,K+1),C(1,J,K))+GK(C(1,J,K-1),C(1,J,K)))+
*    FKB*(B(9,JB,KB)-C(1,J,K)))
        COUT(27,J,K)=FO(3)*(FK(C(27,J,K))*(GK(C(26,J,K),C(27,J,K))+
*    GK(C(27,J+1,K),C(27,J,K))+GK(C(27,J-1,K),C(27,J,K))+
*    GK(C(27,J,K+1),C(27,J,K))+GK(C(27,J,K-1),C(27,J,K)))+
*    FKB*(B(19,JB,KB)-C(27,J,K)))
   5 CONTINUE

*  F. THE TOP AND BOTTOM END EDGES  (MEDIUM INTERFACE)

      KB=1
      JB=NC-1
      JMAX=NC+9
      DO 6 I=2,26
       IB=(I-.5)/3+10
       COUT(I,1,1)=FO(3)*(FK(C(I,1,1))*(GK(C(I+1,1,1),C(I,1,1))+
*    GK(C(I-1,1,1),C(I,1,1))+GK(C(I,2,1),C(I,1,1))+
*    2.*GK(C(I,1,2),C(I,1,1)))+FKB*(B(IB,JB,KB)-C(I,1,1)))+
*    BI(3)*(TINF4-C(I,1,1)**4)+BI(4)*(TINF-C(I,1,1))
       COUT(I,27,1)=FO(3)*(FK(C(I,27,1))*(GK(C(I+1,27,1),C(I,27,1))+
*    GK(C(I-1,27,1),C(I,27,1))+GK(C(I,26,1),C(I,27,1))+
*    2.*GK(C(I,27,2),C(I,27,1)))+FKB*(B(IB,JMAX,KB)-C(I,27,1)))+
*    BI(3)*(TINF4-C(I,27,1)**4)+BI(4)*(TINF-C(I,27,1))
       COUT(I,1,8)=FO(3)*(FK(C(I,1,8))*(GK(C(I+1,1,8),C(I,1,8))+
*    GK(C(I-1,1,8),C(I,1,8))+GK(C(I,2,8),C(I,1,8))+
*    GK(C(I,1,7),C(I,1,8)))+FKB*(B(IB,JB,3)+B(IB,NC,4)
*    -2.*C(I,1,8)))
       COUT(I,27,8)=FO(3)*(FK(C(I,27,8))*(GK(C(I+1,27,8),C(I,27,8))+
*    GK(C(I-1,27,8),C(I,27,8))+GK(C(I,26,8),C(I,27,8))+
*    GK(C(I,27,7),C(I,27,8)))+FKB*(B(IB,JMAX,3)+B(IB,NC+8,4)-
*    2.*C(I,27,8)))
   6 CONTINUE

*  G. THE TOP AND BOTTOM SIDE EDGES (MEDIUM INTERFACE)

      DO 7 J=2,26
       JB=(J-.5)/3+NC
       COUT(1,J,1)=FO(3)*(FK(C(1,J,1))*(GK(C(1,J+1,1),C(1,J,1))+
*    GK(C(1,J-1,1),C(1,J,1))+GK(C(2,J,1),C(1,J,1))+
```

```
     *    2.*GK(C(1,J,2),C(1,J,1)))+FKB*(B(9,JB,1)-C(1,J,1)))+
     *    BI(3)*(TINF4-C(1,J,1)**4)+BI(4)*(TINF-C(1,J,1))
          COUT(27,J,1)=FO(3)*(FK(C(27,J,1))*(GK(C(27,J+1,1),C(27,J,1))+
     *    GK(C(27,J-1,1),C(27,J,1))+GK(C(26,J,1),C(27,J,1))+
     *    2.*GK(C(27,J,2),C(27,J,1)))+FKB*(B(19,JB,1)-C(27,J,1)))+
     *    BI(3)*(TINF4-C(27,J,1)**4)+BI(4)*(TINF-C(27,J,1))
          COUT(1,J,8)=FO(3)*(FK(C(1,J,8))*(GK(C(1,J+1,8),C(1,J,8))+
     *    GK(C(1,J-1,8),C(1,J,8))+GK(C(2,J,8),C(1,J,8))+
     *    GK(C(1,J,7),C(1,J,8)))+FKB*(B(9,JB,3)+B(10,JB,4)
     *    -2.*C(1,J,8)))
          COUT(27,J,8)=FO(3)*(FK(C(27,J,8))*(GK(C(27,J+1,8),C(27,J,8))+
     *    GK(C(27,J-1,8),C(27,J,8))+GK(C(26,J,8),C(27,J,8))+
     *    GK(C(27,J,7),C(27,J,8)))+FKB*(B(19,JB,3)+B(18,JB,4)
     *    -2.*C(27,J,8)))
        7 CONTINUE

*    H. THE CORNER EDGES (MEDIUM INTERFACE)

          DO 8 K=2,7
           KB=2
           IF(K.EQ.2) KB=1
           IF(K.GE.6) KB=3
          COUT(1,1,K)=FO(3)*(FK(C(1,1,K))*(GK(C(2,1,K),C(1,1,K))+
     *    GK(C(1,2,K),C(1,1,K))+GK(C(1,1,K+1),C(1,1,K))+
     *    GK(C(1,1,K-1),C(1,1,K)))+FKB*(B(10,NC-1,KB)+B(9,NC,KB)-
     *    2.*C(1,1,K)))
          COUT(1,27,K)=FO(3)*(FK(C(1,27,K))*(GK(C(2,27,K),C(1,27,K))+
     *    GK(C(1,26,K),C(1,27,K))+GK(C(1,27,K+1),C(1,27,K))+
     *    GK(C(1,27,K-1),C(1,27,K)))+FKB*(B(10,NC+9,KB)+B(9,NC+8,KB)-
     *    2.*C(1,27,K)))
          COUT(27,1,K)=FO(3)*(FK(C(27,1,K))*(GK(C(26,1,K),C(27,1,K))+
     *    GK(C(27,2,K),C(27,1,K))+GK(C(27,1,K+1),C(27,1,K))+
     *    GK(C(27,1,K-1),C(27,1,K)))+FKB*(B(18,NC-1,KB)+B(19,NC,KB)-
     *    2.*C(27,1,K)))
          COUT(27,27,K)=FO(3)*(FK(C(27,27,K))*(GK(C(27,27,K),C(27,27,K))+
     *    GK(C(27,26,K),C(27,27,K))+GK(C(27,27,K+1),C(27,27,K))+
     *    GK(C(27,27,K-1),C(27,27,K)))+FKB*(B(19,NC+9,KB)+
     *    B(18,NC+8,KB)-2.*C(27,27,K)))
        8 CONTINUE

*    I. THE TOP AND BOTTOM CORNERS (MEDIUM INTERFACE)

          COUT(1,1,1)=FO(3)*(FK(C(1,1,1))*(GK(C(2,1,1),C(1,1,1))+
     *    GK(C(1,2,1),C(1,1,1))+GK(C(1,1,2),C(1,1,1)))+FKB*
     *    (B(10,NC-1,1)+B(9,NC,1)-2.*C(1,1,1)))+
     *    BI(3)*(TINF4-C(1,1,1)**4)+BI(4)*(TINF-C(1,1,1))
          COUT(1,27,1)=FO(3)*(FK(C(1,27,1))*(GK(C(2,27,1),C(1,27,1))+
     *    GK(C(1,26,1),C(1,27,1))+GK(C(1,27,2),C(1,27,1)))+FKB*
     *    (B(10,NC+9,1)+B(9,NC+8,1)-2.*C(1,27,1)))+
     *    BI(3)*(TINF4-C(1,27,1)**4)+BI(4)*(TINF-C(1,27,1))
          COUT(27,1,1)=FO(3)*(FK(C(27,1,1))*(GK(C(26,1,1),C(27,1,1))+
     *    GK(C(27,2,1),C(27,1,1))+GK(C(27,1,2),C(27,1,1)))+FKB*
     *    (B(18,NC-1,1)+B(19,NC,1)-2.*C(27,1,1)))+
     *    BI(3)*(TINF4-C(27,1,1)**4)+BI(4)*(TINF-C(27,1,1))
          COUT(27,27,1)=FO(3)*(FK(C(27,27,1))*(GK(C(26,27,1),C(27,27,1))+
     *    GK(C(27,26,1),C(27,27,1))+GK(C(27,27,2),C(27,27,1)))+FKB*
```

```
     *    (B(18,NC+9,1)+B(19,NC+8,1)-2.*C(27,27,1)))+
     *    BI(3)*(TINF4-C(27,27,1)**4)+BI(4)*(TINF-C(27,27,1))
          COUT(1,1,8)=FO(3)*(FK(C(1,1,8))*(GK(C(2,1,8),C(1,1,8))+
     *    GK(C(1,2,8),C(1,1,8))+GK(C(1,1,7),C(1,1,8)))+FKB*
     *    (B(10,NC-1,3)+B(9,NC,3)+B(10,NC,4)-3.*C(1,1,8)))
          COUT(1,27,8)=FO(3)*(FK(C(1,27,8))*(GK(C(2,27,8),C(1,27,8))+
     *    GK(C(1,26,8),C(1,27,8))+GK(C(1,27,7),C(1,27,8)))+FKB*
     *    (B(10,NC+9,3)+B(9,NC+8,3)+B(10,NC+8,4)-3.*C(1,27,8)))
          COUT(27,1,8)=FO(3)*(FK(C(27,1,8))*(GK(C(26,1,8),C(27,1,8))+
     *    GK(C(27,2,8),C(27,1,8))+GK(C(27,1,7),C(27,1,8)))+FKB*
     *    (B(18,NC-1,3)+B(19,NC,3)+B(18,NC,4)-3.*C(27,1,8)))
          COUT(27,27,8)=FO(3)*(FK(C(27,27,8))*(GK(C(26,27,8),C(27,27,8))+
     *    GK(C(27,26,8),C(27,27,8))+GK(C(27,27,7),C(27,27,8)))+FKB*
     *    (B(18,NC+9,3)+B(19,NC+8,3)+B(18,NC+8,4)-3.*C(27,27,8)))

          RETURN
          END
```

E. SOURCE LISTING OF SUBROUTINE MED FORTRAN

```
          SUBROUTINE MED(A,B,C,BOUT)

          DIMENSION A(21,36),BOUT(27,27,10),B(27,27,10),BSUM(27,27,10),
         *C(27,27,8),FO(3),BI(4)
          COMMON NB,NC,YARC,SUM,TINF,BI,FO

*   A. THE INTERIOR NODES

          DO 7 I=2,26
           DO 7 J=2,26
            DO 777 K=2,4
             IF((I.GE.9.AND.I.LE.19.AND.J.GE.NC-1.AND.J.LE.NC+9)
     *       .AND..NOT.(K.EQ.4.AND.(I.EQ.9.OR.I.EQ.19.OR.J.EQ.NC-1.OR.J.EQ.
     *       NC+9)).AND..NOT.((J.EQ.NC-1.OR.J.EQ.NC+9).AND.(I.EQ.9.OR.I.EQ.
     *       19))) GOTO 777
             BOUT(I,J,K)=FO(2)*(B(I+1,J,K)+B(I-1,J,K)+B(I,J-1,K)+B(I,J+1,K)+
     *                B(I,J,K+1)+B(I,J,K-1)-6.*B(I,J,K))
777         CONTINUE
            DO 7 K=5,9
             BOUT(I,J,K)=FO(2)*(B(I+1,J,K)+B(I-1,J,K)+B(I,J-1,K)+B(I,J+1,K)+
     *                B(I,J,K+1)+B(I,J,K-1)-6.*B(I,J,K))
          7 CONTINUE

*   B. THE TOP AND BOTTOM SURFACE NODES

          DO 8 I=2,26
           DO 8 J=2,26
            BOUT(I,J,10)=FO(2)*(B(I+1,J,10)+B(I-1,J,10)+B(I,J+1,10)+
     *          B(I,J-1,10)+2*B(I,J,9)+BI(2)*TINF-(BI(2)+6.)*B(I,J,10))
             IF((I.GE.9.AND.I.LE.19.AND.J.GE.NC-1.AND.J.LE.NC+9)
     *       .AND..NOT.((J.EQ.NC-1.OR.J.EQ.NC+9).AND.(I.EQ.9.OR.I.EQ.
     *       19))) GOTO 8
            BOUT(I,J,1)=FO(2)*(B(I+1,J,1)+B(I-1,J,1)+B(I,J+1,1)+B(I,J-1,1)
     *                +2*B(I,J,2)+BI(2)*TINF-(BI(2)+6.)*B(I,J,1))
          8 CONTINUE
```

```
*  C. THE EXTERIOR END FACES (COARSE INTERFACE)

      DO 9 I=2,26
       IA=INT((I-.5)/3)+7
       DO 9 K=2,9
        BOUT(I,1,K)=FO(2)*(B(I+1,1,K)+B(I-1,1,K)+B(I,2,K)+B(I,1,K+1)+
     *           B(I,1,K-1)+.5*A(IA,NB-1)-5.5*B(I,1,K))
        BOUT(I,27,K)=FO(2)*(B(I+1,27,K)+B(I-1,27,K)+B(I,26,K)+
     *           B(I,27,K+1)+B(I,27,K-1)+.5*A(IA,NB+9)-5.5*B(I,27,K))
    9 CONTINUE

*  D. THE EXTERIOR SIDE FACES (COARSE INTERFACE)

      DO 10 J=2,26
       JA=INT((J-.5)/3)+NB
       DO 10 K=2,9
        BOUT(1,J,K)=FO(2)*(B(2,J,K)+B(1,J+1,K)+B(1,J-1,K)+B(1,J,K+1)+
     *           B(1,J,K-1)+.5*A(6,JA)-5.5*B(1,J,K))
        BOUT(27,J,K)=FO(2)*(B(26,J,K)+B(27,J+1,K)+B(27,J-1,K)+
     *           B(27,J,K+1)+B(27,J,K-1)+.5*A(16,JA)-5.5*B(27,J,K))
   10 CONTINUE

*  E. THE EXTERIOR END EDGES (COARSE INTERFACE)

      DO 11 I=2,26
       IA=INT((I-.5)/3)+7
       BOUT(I,1,1)=FO(2)*(B(I+1,1,1)+B(I-1,1,1)+B(I,2,1)+B(I,1,2)+
     *          .5*A(IA,NB-1)+BI(2)*TINF-(4.5+BI(2))*B(I,1,1))
       BOUT(I,1,10)=FO(2)*(B(I+1,1,10)+B(I-1,1,10)+B(I,2,10)+B(I,1,9)+
     *          .5*A(IA,NB-1)+BI(2)*TINF-(4.5+BI(2))*B(I,1,10))
       BOUT(I,27,1)=FO(2)*(B(I+1,27,1)+B(I-1,27,1)+B(I,26,1)+B(I,27,2)+
     *          .5*A(IA,NB+9)+BI(2)*TINF-(4.5+BI(2))*B(I,27,1))
       BOUT(I,27,10)=FO(2)*(B(I+1,27,10)+B(I-1,27,10)+B(I,26,10)+
     *     B(I,27,9)+.5*A(IA,NB+9)+BI(2)*TINF-(4.5+BI(2))*B(I,27,10))
   11 CONTINUE

*  F. THE COARSE SIDE EDGES (COARSE INTERFACE)

      DO 12 J=2,26
       JA=INT((J-.5)/3)+NB
       BOUT(1,J,1)=FO(2)*(B(2,J,1)+B(1,J+1,1)+B(1,J-1,1)+B(1,J,2)+
     *          .5*A(6,JA)+BI(2)*TINF-(4.5+BI(2))*B(1,J,1))
       BOUT(27,J,1)=FO(2)*(B(26,J,1)+B(27,J+1,1)+B(27,J-1,1)+B(27,J,2)+
     *          .5*A(16,JA)+BI(2)*TINF-(4.5+BI(2))*B(27,J,1))
       BOUT(1,J,10)=FO(2)*(B(2,J,10)+B(1,J+1,10)+B(1,J-1,10)+B(1,J,9)+
     *          .5*A(6,JA)+BI(2)*TINF-(4.5+BI(2))*B(1,J,10))
       BOUT(27,J,10)=FO(2)*(B(26,J,10)+B(27,J+1,10)+B(27,J-1,10)+
     *        B(27,J,9)+.5*A(16,JA)+BI(2)*TINF-(4.5+BI(2))*B(27,J,10))
   12 CONTINUE

*  G. THE EXTERIOR CORNER EDGES (COARSE INTERFACE)

      DO 13 K=2,9
       BOUT(1,1,K)=FO(2)*(B(2,1,K)+B(1,2,K)+B(1,1,K+1)+B(1,1,K-1)+
     *          .5*(A(6,NB)+A(7,NB-1))-5.*B(1,1,K))
       BOUT(27,1,K)=FO(2)*(B(26,1,K)+B(27,2,K)+B(27,1,K+1)+B(27,1,K-1)+
```

```
     *          .5*(A(16,NB)+A(15,NB-1))-5.*B(27,1,K))
          BOUT(1,27,K)=FO(2)*(B(2,27,K)+B(1,26,K)+B(1,27,K+1)+B(1,27,K-1)+
     *          .5*(A(6,NB+8)+A(7,NB+9))-5.*B(1,27,K))
          BOUT(27,27,K)=FO(2)*(B(26,27,K)+B(27,26,K)+B(27,27,K+1)+
     *          B(27,27,K-1)+.5*(A(16,NB+8)+A(15,NB+9))-5.*B(27,27,K))
   13 CONTINUE

*  H. THE EXTERIOR CORNERS (COARSE INTERFACE)

          BOUT(1,1,1)=FO(2)*(B(2,1,1)+B(1,2,1)+B(1,1,2)+.5*(A(6,NB)+
     *          A(7,NB-1))+BI(2)*TINF-(4.+BI(2))*B(1,1,1))

          BOUT(1,1,10)=FO(2)*(B(2,1,10)+B(1,2,10)+B(1,1,9)+.5*(A(6,NB)+
     *          A(7,NB-1))+BI(2)*TINF-(4.+BI(2))*B(1,1,10))
          BOUT(1,27,1)=FO(2)*(B(2,27,1)+B(1,26,1)+B(1,27,2)+.5*(A(6,NB+8)+
     *          A(7,NB+9))+BI(2)*TINF-(4.+BI(2))*B(1,27,1))
          BOUT(1,27,10)=FO(2)*(B(2,27,10)+B(1,26,10)+B(1,27,9)+.5*
     *          (A(6,NB+8)+A(7,NB+9))+BI(2)*TINF-(4.+BI(2))*B(1,27,10))
          BOUT(27,1,1)=FO(2)*(B(26,1,1)+B(27,2,1)+B(27,1,2)+.5*(A(16,NB)+
     *          A(15,NB-1))+BI(2)*TINF-(4.+BI(2))*B(27,1,1))
          BOUT(27,1,10)=FO(2)*(B(26,1,10)+B(27,2,10)+B(27,1,9)+.5*
     *          (A(16,NB)+A(15,NB-1))+BI(2)*TINF-(4.+BI(2))*B(27,1,10))
          BOUT(27,27,1)=FO(2)*(B(26,27,1)+B(27,26,1)+B(27,27,2)+.5*
     *          (A(16,NB+8)+A(15,NB+9))+BI(2)*TINF-(4.+BI(2))*B(27,27,1))
          BOUT(27,27,10)=FO(2)*(B(26,27,10)+B(27,26,10)+B(27,27,9)+.5*
     *         (A(16,NB+8)+A(15,NB+9))+BI(2)*TINF-(4.+BI(2))*B(27,27,10))

*  I. THE BOTTOM OF THE EXCLUSION ZONE (FINE INTERFACE)

          DO 20 I=10,18
           DO 20 J=NC,NC+8
            TC=0.
            DO 21 IC=3*(I-10)+1,3*(I-10)+3
             DO 21 JC=3*(J-NC)+1,3*(J-NC)+3
   21        TC=TC+C(IC,JC,8)
           BOUT(I,J,4)=FO(2)*(B(I+1,J,4)+B(I-1,J,4)+B(I,J+1,4)+B(I,J-1,4)
     *          +B(I,J,5)+TC/6.-6.5*B(I,J,4))
   20 CONTINUE

*  J.  THE SIDE OF THE EXCLUSION ZONE (FINE INTERFACE)

          DO 22 J=NC,NC+8
           DO 22 K=2,3
            TC1=0.
            TC2=0.
            DO 23 JC=3*(J-NC)+1,3*(J-NC)+3
             DO 23 KC=3*(K-2)+3,3*(K-2)+5
             TC1=TC1+C(1,JC,KC)
   23        TC2=TC2+C(27,JC,KC)
           BOUT(9,J,K)=FO(2)*(B(8,J,K)+B(9,J+1,K)+B(9,J-1,K)+B(9,J,K+1)+
     *          B(9,J,K-1)+TC1/6.-6.5*B(9,J,K))
          BOUT(19,J,K)=FO(2)*(B(20,J,K)+B(19,J+1,K)+B(19,J-1,K)+
     *          B(19,J,K+1)+B(19,J,K-1)+TC2/6.-6.5*B(19,J,K))
   22 CONTINUE

*  K. THE ENDS OF THE EXCLUSION ZONE (FINE INTERFACE)
```

```
      DO 24 I=10,18
       DO 24 K=2,3
        TC1=0.
        TC2=0.
        DO 25 IC=3*(I-10)+1,3*(I-10)+3
         DO 25 KC=3*(K-2)+3,3*(K-2)+5
         TC1=TC1+C(IC,1,KC)
25       TC2=TC2+C(IC,27,KC)
        BOUT(I,NC-1,K)=FO(2)*(B(I+1,NC-1,K)+B(I-1,NC-1,K)+B(I,NC-2,K)
     *      +B(I,NC-1,K+1)+B(I,NC-1,K-1)+TC1/6.-6.5*B(I,NC-1,K))
        BOUT(I,NC+9,K)=FO(2)*(B(I+1,NC+9,K)+B(I-1,NC+9,K)+B(I,NC+10,K)
     *      +B(I,NC+9,K+1)+B(I,NC+9,K-1)+TC2/6.-6.5*B(I,NC+9,K))
24    CONTINUE

*     L.  THE SIDE EDGE OF THE EXCLUSION ZONE (FINE INTERFACE)

      DO 26 J=NC,NC+8
       TC1=0.
       TC2=0.
       DO 27 JC=3*(J-NC)+1,3*(J-NC)+3
        TC1=TC1+C(1,JC,1)+2*C(1,JC,2)
27      TC2=TC2+C(27,JC,1)+2*C(27,JC,2)
       BOUT(9,J,1)=FO(2)*(B(8,J,1)+B(9,J+1,1)+B(9,J-1,1)+B(9,J,2)+
     *         TC1/6.+BI(2)*TINF-(5.5+BI(2))*B(9,J,1))
       BOUT(19,J,1)=FO(2)*(B(20,J,1)+B(19,J+1,1)+B(19,J-1,1)+B(19,J,2)+
     *         TC2/6.+BI(2)*TINF-(5.5+BI(2))*B(19,J,1))
26    CONTINUE

*     M.  THE END EDGE OF THE EXCLUSION ZONE (FINE INTERFACE)

      DO 28 I=10,18
       TC1=0.
       TC2=0.
       DO 29 IC=3*(I-10)+1,3*(I-10)+3
        TC1=TC1+C(IC,1,1)+2*C(IC,1,2)
29      TC2=TC2+C(IC,27,1)+2*C(IC,27,2)
       BOUT(I,NC-1,1)=FO(2)*(B(I+1,NC-1,1)+B(I-1,NC-1,1)+B(I,NC-2,1)
     *      +B(I,NC-1,2)+TC1/6.+BI(2)*TINF-(5.5+BI(2))*B(I,NC-1,1))
       BOUT(I,NC+9,1)=FO(2)*(B(I+1,NC+9,1)+B(I-1,NC+9,1)+B(I,NC+10,1)
     *      +B(I,NC+9,2)+TC2/6.+BI(2)*TINF-(5.5+BI(2))*B(I,NC+9,1))
28    CONTINUE

      RETURN
      END
```

F. SOURCE LISTING OF SUBROUTINE COR FORTRAN

```
      SUBROUTINE COR(A,B,AOUT)

      REAL A,B,AOUT
      DIMENSION A(21,36),AOUT(21,36),B(27,27,10),FO(3),BI(4)
      COMMON NB,NC,YARC,SUM,TINF,BI,FO

*     A.  THE CORNERS
```

```
      AOUT(1,1)=0.
      AOUT(1,36)=0.
      AOUT(21,1)=0.
      AOUT(21,36)=0.

*  B. THE ENDS

      DO 1 I=2,20
       AOUT(I,1)=FO(1)*(A(I+1,1)+A(I-1,1)+A(I,2)+TINF*BI(1)-(3+BI(1))*
     *           A(I,1))
       AOUT(I,36)=FO(1)*(A(I+1,36)+A(I-1,36)+A(I,35)+TINF*BI(1)-(3+BI(1)
     *          )*A(I,36))
     1 CONTINUE

*  C. THE SIDES

      DO 2 J=2,35
       AOUT(1,J)=FO(1)*(A(1,J+1)+A(1,J-1)+A(2,J)+TINF*BI(1)-(3+BI(1))*
     *           A(1,J))
       AOUT(21,J)=FO(1)*(A(21,J+1)+A(21,J-1)+A(20,J)+TINF*BI(1)-(3+
     *           BI(1))*A(21,J))
     2 CONTINUE

*  D. THE MID POINTS

      NMIN=NB-1
      NMAX=NB+9
      DO 3 I=2,20
       DO 3 J=2,35
        IF (J.GE.NMIN.AND.J.LE.NMAX.AND.I.GE.7.AND.I.LE.15) GOTO 3
        IF (J.GE.NB.AND.J.LE.(NB+8).AND.(I.EQ.6.OR.I.EQ.16)) GOTO 3
        AOUT(I,J)=FO(1)*(A(I+1,J)+A(I-1,J)+A(I,J+1)+A(I,J-1)+BI(1)*TINF
     *            -(4+BI(1))*A(I,J))
     3 CONTINUE

*  E. THE MEDIUM ENDS TRANSITION

      DO 4 I=7,15
       TA=0.
       TB=0.
       DO 5 IB=(I-7)*3+1,(I-6)*3
        TA=B(IB,1,1)/2.+TA
        TA=B(IB,1,10)/2.+TA
        TB=B(IB,27,1)/2.+TB
        TB=B(IB,27,10)/2.+TB
        DO 5 K=2,9
         TA=B(IB,1,K)+TA
     5   TB=B(IB,27,K)+TB
       AOUT(I,NMIN)=FO(1)*(A(I+1,NMIN)+A(I-1,NMIN)+A(I,NMIN-1)+TA/18.+
     *              BI(1)*TINF-(4.5+BI(1))*A(I,NMIN))
       AOUT(I,NMAX)=FO(1)*(A(I+1,NMAX)+A(I-1,NMAX)+A(I,NMAX+1)+TB/18.+
     *              BI(1)*TINF-(4.5+BI(1))*A(I,NMAX))
     4 CONTINUE

*  F. THE MEDIUM SIDE TRANSITION
```

```
      DO 6 J=NB,NB+8
       TA=0.
       TB=0.
       DO 7 JB=(J-NB)*3+1,(J-NMIN)*3
        TA=B(1,JB,1)/2.+TA
        TA=B(1,JB,10)/2.+TA
        TB=B(27,JB,1)/2.+TB
        TB=B(27,JB,10)/2.+TB
        DO 7 K=2,9
         TA=B(1,JB,K)+TA
    7    TB=B(27,JB,K)+TB
      AOUT(6,J)=FO(1)*(A(5,J)+A(6,J-1)+A(6,J+1)+TA/18.+
     *               BI(1)*TINF-(4.5+BI(1))*A(6,J))
      AOUT(16,J)=FO(1)*(A(17,J)+A(16,J-1)+A(16,J+1)+TB/18.+
     *               BI(1)*TINF-(4.5+BI(1))*A(16,J))
    6 CONTINUE
      RETURN
      END
```

APPENDIX B. STABILITY OF RUNGE-KUTTA

Applying fourth order Runge-Kutta to the semi-discrete, explicit finite difference equations has an effect on their stability. The explicit technique has a three dimensional stability criterion of (1-6Fo) $\geq$ 0, or

$$Fo \leq \frac{1}{6} \tag{B.1}$$

For prescribed values of Δx and α, this criterion determines the upper bound on Δt. The stability criterion was determined by collecting all of the $T^n_{i,j,k}$ terms, to obtain the form of the coefficient.

A similar expression can be determined by gathering the terms of $T^n_{i,j,k}$ after the completion of Runge-Kutta. This requires a good deal of algebra since a fourth order polynomial is the result. The general form of Runge-Kutta is shown in equation B.2.

$$\begin{aligned} K1 &= f(T^n) \\ K2 &= f(T^n + \frac{K1}{2}) \\ K3 &= f(T^n + \frac{K2}{2}) \\ K4 &= f(T^n + K3) \\ T^{n+1} &= T^n + \frac{(K1 + 2(K2 + K3) + K4)}{6} \end{aligned} \tag{B.2}$$

Because of the amount of notation required to derive the coefficient of $T^n_{i,j,k}$, the i,j,k subscript will be dropped and the different terms will be indicated by a relative subscript, where 000 is the same as i,j,k and 100 is i+1,j,k, and so forth. Substituting the finite difference equation for f into equation B.2 and using this notation gives

$$\begin{aligned} K1_{000} &= Fo(T_{100} + T_{-100} + T_{010} + T_{0-10} + T_{001} + T_{00-1} - 6T_{000}) \\ K2_{000} &= K1_{000} + \frac{Fo}{2}(K1_{100} + K1_{-100} + K1_{010} + K1_{0-10} + K1_{001} + K1_{00-1} - 6K1_{000}) \\ K3_{000} &= K1_{000} + \frac{Fo}{2}(K2_{100} + K2_{-100} + K2_{010} + K2_{0-10} + K2_{001} + K2_{00-1} - 6K2_{000}) \\ K4_{000} &= K1_{000} + Fo(K3_{100} + K3_{-100} + K3_{010} + K3_{0-10} + K3_{001} + K3_{00-1} - 6K3_{000}) \end{aligned} \tag{B.3}$$

A difference between the form of equation B.2 and equation B.3 is that, in this case, one can take advantage of f for a finite difference equation being a linear operator. This property of $f(T+K)=f(T)+f(K)$ allows greatly simplifying the algebra.

Now all that is required is to group the terms of T_{000} and find the coefficient. The simplest is the K1 term, which from equation B.3 is seen to have a coefficient of

$$\begin{aligned} K1_{000} &\rightarrow -6Fo \\ K1_{100} &\rightarrow 1Fo \end{aligned} \qquad (B.4)$$

The $K1_{100}$ is the case general. Any of the other five adjacent differences, -100, 010, 0-10, 001 and 00-1, will have the same coefficient of T_{000} as 100. This simplifies finding the coefficients for the rest of the terms.

For K2 it becomes a little more difficult. The contribution of $K1_{000}$ is already known. The values for $K1_{100}$ type terms is given in equation B.4, which yields,

$$\begin{aligned} K2_{000} &\rightarrow -6Fo + \frac{Fo}{2}(Fo+Fo+Fo+Fo+Fo+Fo-6(-6Fo)) \\ &\rightarrow -6Fo + 21Fo^2 \end{aligned} \qquad (B.5)$$

The next term to expand is K3. This will be done in several parts, the $K1_{000}$ and $K2_{000}$ terms are already known. Equation B.6 shows the expansion for a $K2_{100}$ term.

$$\begin{array}{ccccccccc} K2_{100} = K1_{100} + \frac{Fo}{2}(K1_{200} & + K1_{000} & + K1_{110} & + K1_{1-10} & + K1_{101} & + K1_{10-1} & - 6K1_{100}) \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ Fo & 0 & -6Fo & 0 & 0 & 0 & 0 & -6Fo \end{array} \qquad (B.6)$$

$$K2_{100} \rightarrow Fo - 6Fo^2$$

Since there are six terms of the form $K2_{100}$ the total coefficient for K3 is,

$$\begin{aligned} K3_{000} &\rightarrow -6Fo + \frac{Fo}{2}(6(Fo-6Fo^2) - 6(-6Fo+21Fo^2)) \\ &\rightarrow -6Fo + 21Fo^2 - 81Fo^3 \end{aligned} \qquad (B.7)$$

The final term is K4, which has a long expansion. Again it will be broken into parts. The $K1_{000}$ and $K3_{000}$ terms are already known from the above. It is necessary to expand the $K3_{100}$ terms, as shown below.

$$K3_{100} = K1_{100} + \frac{Fo}{2}(K2_{200} + K2_{000} + K2_{110} + K2_{1-10} + K2_{101} + K2_{10-1} - 6K2_{100}) \quad (B.8)$$
$$\text{I} \qquad \text{II} \quad \text{III} \quad \text{IV} \quad \text{IV} \quad \text{IV} \quad \text{IV} \quad \text{V}$$

Equation B.8 has five types of terms, as identified by the Roman numerals. The type I, type III and type V terms are already defined by equations B.4 and B.7. Expanding the type II term gives,

$$K2_{200} = K1_{200} + \frac{Fo}{2}(K1_{300} + K1_{100} + K1_{210} + K1_{2-10} + K1_{201} + K1_{20-1} - 6K1_{200}) \quad (B.9)$$
$$\downarrow \qquad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow$$
$$0 \qquad 0 \quad Fo \quad 0 \quad 0 \quad 0 \quad 0 \quad 0$$

$$K2_{200} \rightarrow \frac{Fo^2}{2}$$

Expanding the type IV term gives,

$$K2_{110} = K1_{110} + \frac{Fo}{2}(K1_{210} + K1_{010} + K1_{120} + K1_{100} + K1_{111} + K1_{11-1} - 6K1_{110}) \quad (B.10)$$
$$\downarrow \qquad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow$$
$$0 \qquad 0 \quad Fo \quad 0 \quad Fo \quad 0 \quad 0 \quad 0$$

$$K2_{110} \rightarrow Fo^2$$

Combining equations B.4, B.7, B.8, B.9 and B.10 gives,

$$\begin{aligned} K3_{100} &\rightarrow Fo + \frac{Fo}{2}\left[\frac{Fo^2}{2} - 6Fo + 21Fo^2 + 4(Fo^2) - 6(Fo - 6Fo^2)\right] \\ &\rightarrow Fo - 6Fo^2 + \frac{123Fo^3}{4} \end{aligned} \quad (B.11)$$

The coefficient for the terms in K4 of equation B.3 can now be found by using equations B.4, B.7 and B.11. This gives,

$$\begin{aligned} K4_{000} &\rightarrow -6Fo + Fo\left[6(Fo - 6Fo^2 + \frac{123Fo^3}{4}) - 6(-6Fo + 21Fo^2 - 81Fo^3)\right] \\ &\rightarrow -6Fo + 42Fo^2 - 162Fo^3 + \frac{1341}{2}Fo^4 \end{aligned} \quad (B.12)$$

The final step is to gather all of the terms from equation B.2 into the Runge-Kutta sum of equation B.1. This results in the coefficient of $T^n_{i,j,k}$, and is given in equation B.13.

$$\rightarrow 1 - 6Fo + 21Fo^2 - 54Fo^3 + \frac{447}{4} Fo^4 \qquad (B.13)$$

This fourth order polynomial must satisfy two conditions for stability, the traditional explicit, that it be greater than zero and an additional condition, that it be less than one. The traditional limit prevents oscillations about a solution which violates the laws of thermodynamics. The second limit is based on the law of heat conduction, for if the coefficient is greater than one, it is possible for the temperature at a point to rise, even though every other point is at a lower temperature.

The above limits the maximum value of the Fourier number for a three dimensional problem to 0.37, which is better than the normal value of one sixth by over a factor of two. While not a limit, operating above the minimum point of the Runge-Kutta curve, which occurs at a Fourier number of about 0.22, causes a rapid loss of accuracy. This should be apparent, for at this point, taking a larger time step causes smaller temperature changes instead of the expected larger. A plot of the temperature coefficient versus Fo is shown in Figure 36, where it is interesting to note that, for Runge-Kutta, the coefficient never drops below zero. This implies that for a three dimensional problem using Runge-Kutta, taking too large a time step will not cause solution oscillation, but rapid increase in error. Also shown for comparison is the coefficient for the implicit method. Even though this method is unconditionally stable, it does not have greater accuracy for the Fourier numbers where the other two methods are stable.

$$\rightarrow 1 - 2Fo + 3Fo^2 - \frac{10}{3} Fo^3 + \frac{35}{12} Fo^4 \qquad (B.14)$$

$$\rightarrow 1 - 4Fo + 10Fo^2 - \frac{56}{3} Fo^3 + \frac{169}{6} Fo^4 \qquad (B.15)$$

Similar calculations may also be performed for one and two dimensions. These results are shown in equations B.14 and B.15, respectively. The comparisions of the temperature coefficients versus Fourier number are shown in Figure 37 for the one dimensional and Figure 38 for the two dimensional finite difference equations. The use of Runge-Kutta improves the stability of the explicit finite difference in all cases, however the improvement is greater the higher the dimensions.

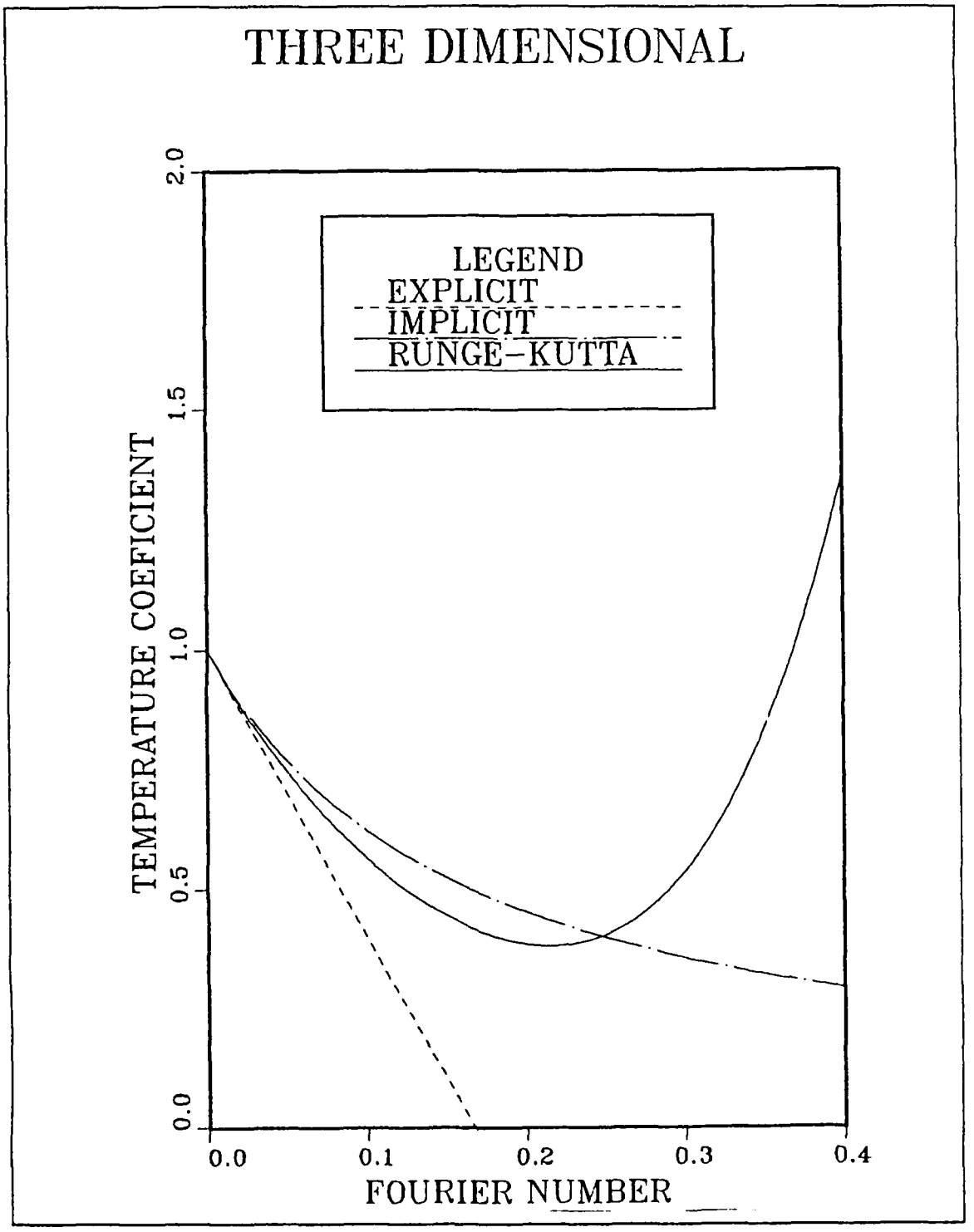


Figure 36. Temperature Coefficient Versus Fourier Number for Three Dimesions

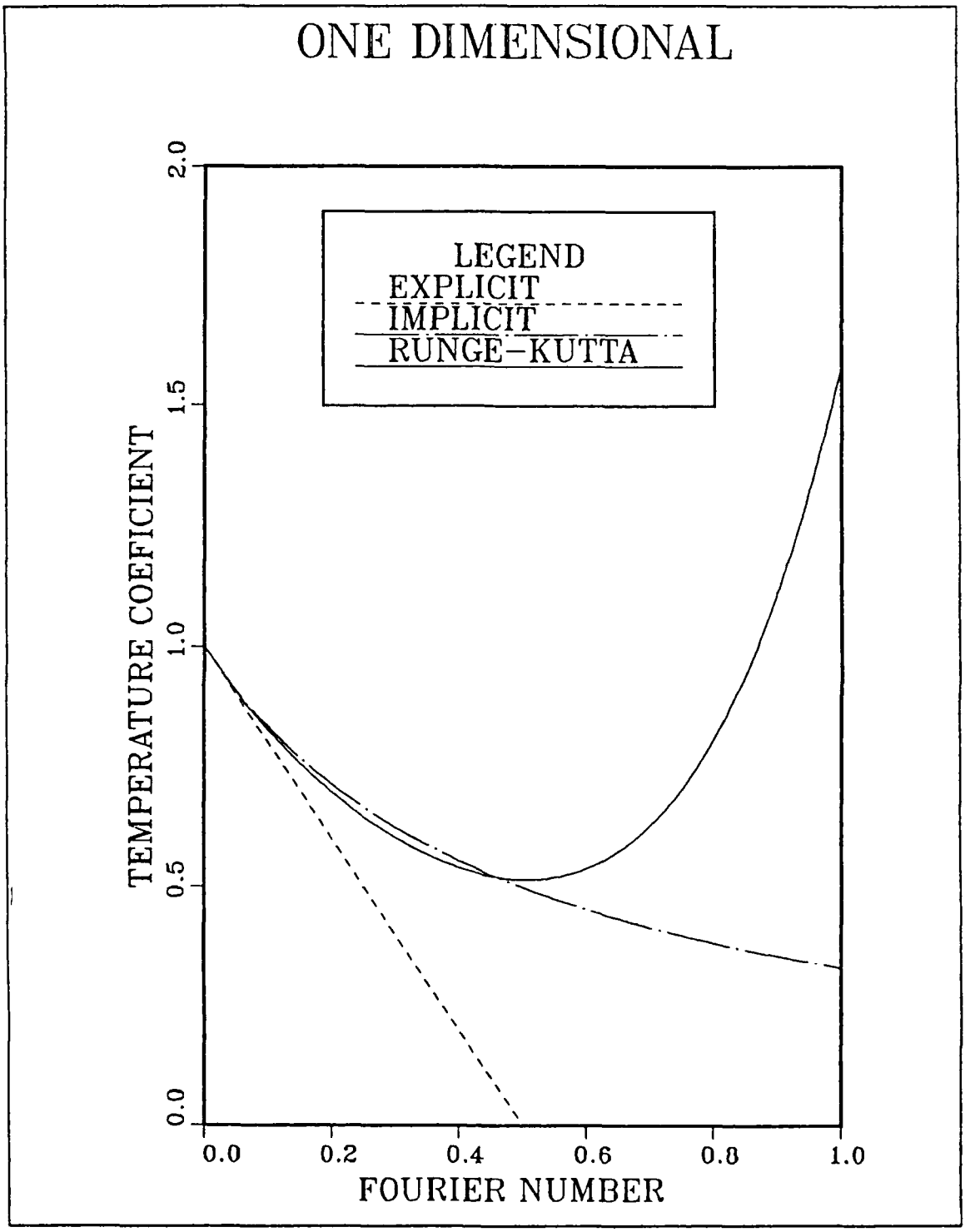


Figure 37. Temperature Coefficient Versus Fourier Number for One Dimension

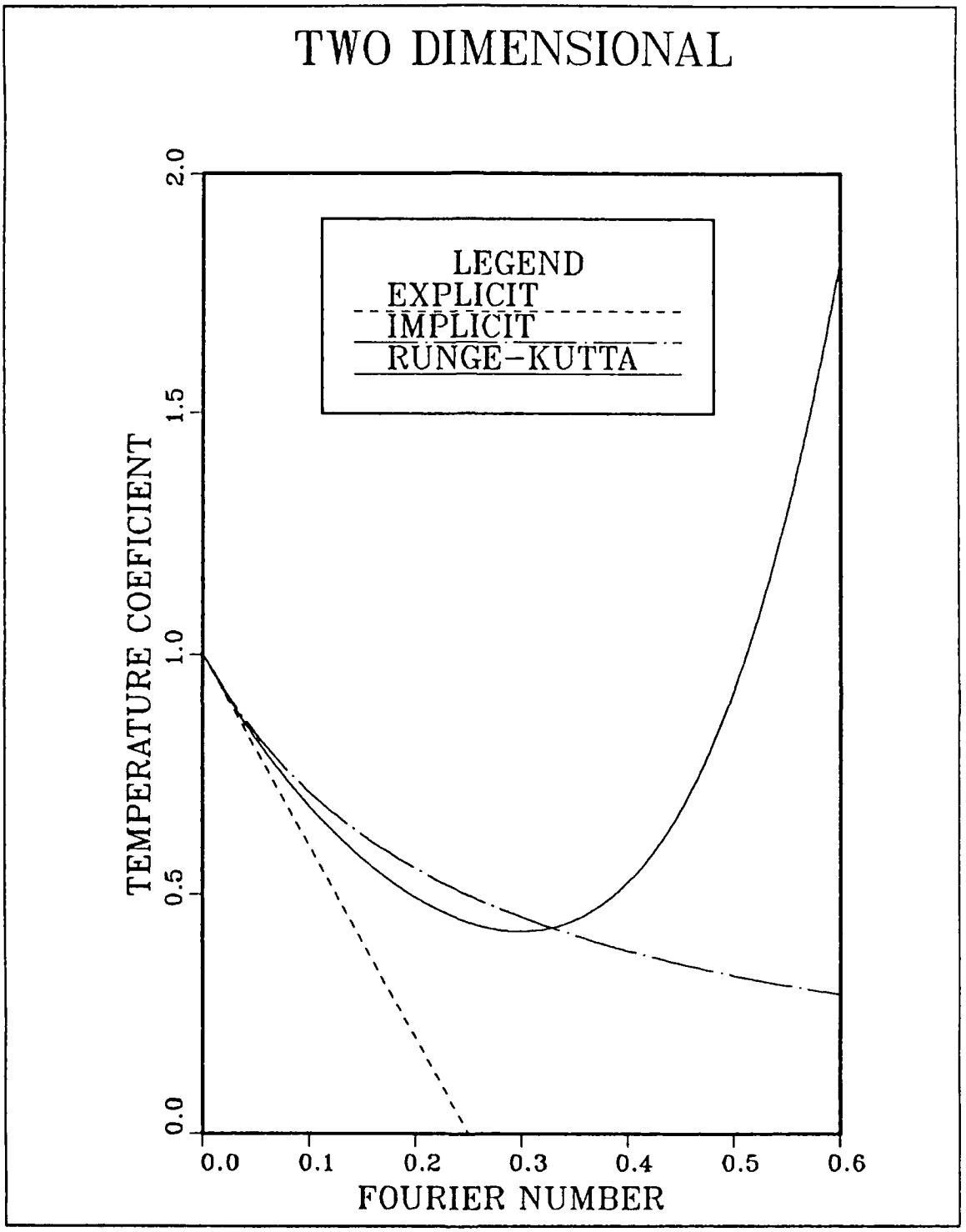


Figure 38. Temperature Coefficient Versus Fourier Number for Two Dimensions

APPENDIX C. SOURCE LISTING AND EXPLANATION OF AUXILIARY COMPUTER PROGRAMS

A. PLOTTING PROGRAMS

There is one major plotting program, named PLOT, and several variants which have been modified to handle particular cases. The source listing that follows is the most general. This program is designed to read the binary output files produced by the weld modeling programs and provide either contour or three dimensional drawings of the temperatures along a plane slicing the weld area. Since this program grew as the need arose, it is not well organized. However, it is an interactive program and can be easily used.

There are three basic presentations available. A temperature contour at a horizontal surface at any depth, a temperature three dimensional plot at a horizontal surface at any depth and a contour of a vertical surface at any X location. A secondary option is to draw temperature profiles for comparison, which was only used when validating the weld model.

1. User Guide

To use this program, you first must have available the output files from running the program WELD, WELDLF or WELDMA. Insure that you have at least 1500K of storage by typing GETSTOR 1500K in CMS. Your profile exec should include the NONIMSL library, since one routine is called from this library. Then type DISSPLA PLOT, to execute this DISSPLA program. After typing enter in response to the question about file definitions, the program will be loaded.

There are two ways to display the output of the program, the first is onto the screen of a tektronics and the second is to create a metafile and then display the output using DISSPOP. The later is done by choosing the COMPRS option, which then allows using the laser printer for higher quality graphics. The next question has you select 3D plots or profiles, in general you will select 3D plots. You will next be asked the contour interval, 500 is a good typical value. At this point you are in the program loop so you can try another value later. The last preparatory question allows you to add a specific label to the graph. This label can be 60 characters long and should be ended with a '$' or else the title will not be centered.

The next choice is which data you desire to see. The first three choices are taken from files which saved a specific surface at half second intervals. The last three choices are from the final data saved and hold the entire temperature matrices and either horizontal X-Y planes or vertical X-Z planes may be looked at. If you select a running surface, you will be asked for a time. Since there is one surface saved every 0.5 seconds of simulation, you can ask to see any of these. If you attempt to look at a surface that hasn't been saved yet, then the program will fail. For the last three choices, you will have to specify the type of plot desired and the location of the plane to plot. As indicated in the question, these locations are always referenced to the nodes of the surface you are looking at. If you are looking at an X-Y surface you will be asked if you want a 3D plot. A 3D plot draws the temperature as a third dimension over the surface, instead of a contour map.

If you have selected Tektronics, the plot will now be displayed. You can then choose to draw another plot. If you have selected compress, a metafile is created. After you have made all of the plots you desire, you can use DISSPOP to send your plots any where you choose. Just type DISSPOP and enter and then answer the prompts.

2. Program Description

The program uses the package of graphics programs called DISSPLA [Ref. 20] and this discussion assumes that the reader is familiar with these routines. The first section of the program defines the titles that are placed on the head of the graphs. By use of an equivalence statement and internal formatted writes these titles are modified as needed to support correctly identifying each graph.

The second portion of the program inputs the major options selected by the user. The first choice is to use a tektronics or to use DISSPOP, the later being run after this program, using the metafiles created by the COMPRS subroutine. The second major option is to create three dimensional temperature profiles or to compare line profiles. This later choice is a simple routine which makes use of the NONIMSL routine PLOTD. For normal plotting the next choice will be the contour interval. The contour interval should be large enough to keep three dimensional graphs on scale and contour plots to not have an excessive number of lines. Experience has shown that 500 degrees is a good value. The next prompt allows inputting a fourth line of title to include additional information about this picture, such as arc power, speed, etc. The last question of the input section asks which type of plot is desired.

Since the Fine and Medium zone horizontal surfaces are both square, the program is written with them using the same calls. Since the scale is different, the first step

is to set the scale variable. This is only used if a Fine or Medium horizontal plot is being made.

The next section is a large IF-THEN-ELSE statement which handles the two major cases, running output or final output, for graphing. The first half is running output, so the user is asked at what time the output is desired, a time which will always be truncated to the nearest 0.5 interval of time. If either a Fine or Medium surface is to be read, then the file SURF is read until the appropriate surface is inputted. If the X-Z profile is to be read, then the file CUT is read until the appropriate vertical surface is inputted. In this last case, the matrix is read in reverse order for the Z direction to allow proper orientation on output by the contour routine. The variable IMAT is used to identify the type of output geometry to use later in the program. At this point the AREA2D DISSPLA subroutine is called to define the plot area.

The second half of the IF-THEN-ELSE statement handles the three cases where the FINAL output will be used to create the plots. The three dimensional arrays (two dimensional for the Coarse zone) are read in. If the Coarse zone is chosen, there is only one surface since it is two dimensional, and the AREA2D is called and IMAT set. For the Fine and Medium zones, either the horizontal X-Y plane or vertical X-Z plane must be selected for plotting. Next, the location of the plane must be specified, that is Z for the X-Y plane and Y for the X-Z plane. This value is in nodal position, not physical distance. The program will convert nodal position to distance when drawing the plot. Again IMAT is set for the type of plot and AREA2D is called to set the plotting area.

At this point the IF-THEN-ELSE block has ended and the option of plotting a three dimensional profile versus a contour plot is given for horizontal Fine or Medium surfaces. If a contour plot is chosen the sequence of DISSPLA subroutines is called to produce the desired plot. The four types of surfaces that can be plotted are determined by the variable IMAT, which is specified in the variable definitions at the start of the program. When the contour plot is done, the user is allowed the option of creating another plot. If for the Fine or Medium horizontal plots the three dimensional plot was chosen, then the DISSPLA routines for this plot are called. Again the program allows the option of producing another graph.

The last section of the program is for the special case of producing comparative temperature profiles in the fine zone from several different sources. The input files should be the Final files from running the Weld programs or the Rosenthal verification program. These are binary files and should have different names. Either the X or the Y axis may be chosen for the graph. This same axis will be used for all graphs. It is

possible to plot several temperature profiles from the same file, all at different Y-Z (X-Z) locations. Upon completion, the program loops to allow producing additional plots.

a. Variants

There are two major variants to the plotting program. There is a modification, PLOTL, for plotting the larger arrays associated with the WELDC program which is used for cooling rate calculations. The only difference in this program is that the Fine and Medium arrays have been enlarged and the indices changed were appropriate. There is a modification, PLOTYZ, for plotting Y-Z vertical planes vice the standard X-Z vertical planes. The only difference in this program is that for Final Fine and Medium plots, the Y-Z plane can be selected vice the X-Z plane.

3. Source Listing of Program PLOT

```
      PROGRAM PLOT
************************************************************************
*  THIS PROGRAM PLOTS CONTOURS OF THE OUTPUT DATA FROM RUNNING THE     *
*  PROGRAM WELD.  THE FOLLOWING VARIABLES ARE DEFINED:                 *
*  T    :  THE TEMPERATURE DIFFERENCE BETWEEN CONTOURS                 *
*  N1   :  SPECIFIES WHICH PLOT HEADING IS FIRST                       *
*  N2   :  SPECIFIES WHICH PLOT HEADING IS SECOND                      *
*  N    :  LIST OF LENGTHS OF THE HEADINGS                             *
*  HEAD :  THE LIST OF PLOT HEADINGS                                   *
*  HD,DEPTH,TIME4,TIME8,H9Y,H10Y ARE ALL USED TO MODIFY HEADINGS       *
*  IMAT :  INTEGER SPECIFYING PLOT TYPE BEING OUTPUT (DIMENSIONS)      *
*          VALUE     PLOT TYPE                ASSOCIATED MATRIX      *
*            1       XY PLOT FOR FINE OR MEDIUM       ZMAT           *
*            2       XZ PLOT FOR FINE                 XZF            *
*            3       XY PLOT FOR COARSE               ZCMAT          *
*            4       XZ PLOT FOR MEDIUM               XZM            *
************************************************************************

      DIMENSION ZMAT(27,27),XZF(27,8),ZCMAT(21,36),XZM(27,10),N(10)
     *,C(27,27,8),B(27,27,10),XZFH(27,8),XDATA(27),YDATA(27)
      CHARACTER HEAD(12)*50,TIME4*4,TIME8*4,DEPTH*2,H9Y*3,H10Y*3,H(50)
     *,HD*50,ANS,TEMP*5,USERHD*60,AXS,FNAME*8,XLABEL*60,YLABEL*60,
     *LGDTXT*20
      EQUIVALENCE (HD,H(1)),(H(8),TIME4),(H(23),TIME8),(H(9),DEPTH)
     *,(H(40),H9Y),(H(42),H10Y),(H(37),TEMP)
      COMMON WRK(20000)
      DATA N/29,31,33,19,21,23,23,34,45,47/,HEAD/
     *'FINE ZONE SURFACE TEMPERATURE','MEDIUM ZONE SURFACE TEMPERATURE',
     *'PROFILE OF X-Z TEMPERATURE AT ARC','TIME =    .  SECONDS',
     *'FINE ZONE TEMPERATURE','MEDIUM ZONE TEMPERATURE',
     *'COARSE ZONE TEMPERATURE','DEPTH =    MM  TIME =    .  SECONDS',
     *'FINE PROFILE OF X-Z TEMPERATURE AT Y =     MM',
     *'MEDIUM PROFILE OF X-Z TEMPERATURE AT Y =     MM',
     *'TEMPERATURE BETWEEN CONTOUR LINES =    .  DEG K',
     *'FINE ZONE TEMPERATURE PROFILES$'/

C  OFFER OPTIONS OF DISPLAY TO USER
```

```
      WRITE(*,'(1X,A)')
     *'      THIS PROGRAM DISPLAYS THE OUTPUT FROM THE PROGRAM WELD.',
     *'THE OUTPUT CAN BE DIRECTED TO A TEK618 PLOTTER OR TO A COMPRS',
     *'FILE FOR LATER OUTPUT.  THE OUTPUT ARE CONTOUR PLOTS OF THE',
     *'FINE SURFACE, THE MEDIUM SURFACE OR THE XZ PROFILE AT THE ARC',
     *'AT .5 SECOND INTERVALS OR ANY POSSIBLE HORIZONTAL OR XZ SLICE',
     *'OF THE FINAL TEMPERATURE DISTRIBUTION.  FOR FINE AND MEDIUM',
     *'HORIZONTAL SURFACES IT IS POSSIBLE TO DRAW 3D PLOTS.  FOR THE',
     *'FINE ZONE ONLY IT IS POSSIBLE TO DRAW COMPARATIVE TEMPERATURE',
     *'PROFILES.',' ',
     *'PLEASE SELECT PLOTTING DEVICE: (1) TEXTRONIX 618 (2) COMPRS'
      READ (*,*) IPLOT
      IF (IPLOT.EQ.1) CALL TEK618
      IF (IPLOT.NE.1) CALL COMPRS
    1 WRITE(*,*)' DO YOU WANT: (1) 3D  OR (2) PROFILES?'
      READ(*,*) L
      IF (L.EQ.2) GOTO 1000
      WRITE(*,'(/,1X,A)') 'WHAT IS THE CONTOUR INTERVAL?'
      READ(*,*) T
      HD=HEAD(11)
      WRITE(TEMP,'(F5.1)') T
      T=T/4.
      HEAD(11)=HD
      WRITE(*,*)' TYPE IN ANY ADDITIONAL GRAPH LABEL DESIRED, TERMINATE'
      WRITE(*,*)' WITH A $. (60 CHARACTERS MAX)'
      READ(*,'(A60)')USERHD
      WRITE(*,'(1X,A)') ' ',
     *'SELECT ONE OF THE FOLLOWING OPTIONS:',
     *'(1)  FINE SURFACE AT .5 SECOND INTERVAL',
     *'(2)  MEDIUM SURFACE AT .5 SECOND INTERVAL',
     *'(3)  X-Z PROFILE OF ARC AT .5 SECOND INTERVAL',
     *'(4)  FINAL FINE ZONE TEMPERATURE',
     *'(5)  FINAL MEDIUM ZONE TEMPERATURE',
     *'(6)  FINAL COARSE ZONE TEMPERATURE'
      READ(*,*) IPLOT
      IF (IPLOT.EQ.1.OR.IPLOT.EQ.4) THEN
       SCALE=1.
      ELSE
       SCALE=3.
      ENDIF

C  INPUT THE DATA FOR THE OPTION SELECTED

      IF (IPLOT.LE.3) THEN
       N1=IPLOT
       N2=4
       WRITE(*,*) ' WHAT IS THE DESIRED TIME?'
       READ(*,*) TIME
       TIME=(INT(2*TIME)/2.)
       HD=HEAD(4)
       WRITE(TIME4,'(F4.1)')TIME
       ISKIP=TIME/.5-1
       IF (IPLOT.LE.2) THEN
C  SURFACE PLOT
        OPEN(1,FILE='SURF',STATUS='OLD',FORM='UNFORMATTED')
C  SKIP OVER EARLIER TIME DATA IN FILE
```

```
      DO 10 I=1,ISKIP
       READ(1) ZMAT
10     READ(1) ZMAT
      READ(1) ZMAT
      IF (IPLOT.EQ.2) READ(1)ZMAT
      CALL AREA2D(7.,7.)
      IMAT=1
     ELSE
C  FINE XZ PROFILE AT SPECIFIED TIME
      OPEN(1,FILE='CUT',STATUS='OLD',FORM='UNFORMATTED')
      DO 11 I=1,ISKIP+1
11    READ(1) XZFH
      DO 111 I=1,27
       DO 111 K=1,8
111    XZF(I,K)=XZFH(I,9-K)
      IMAT=2
      CALL AREA2D(7.1,2.4)
     ENDIF
    ELSE
C  PLOT CONTOURS FROM DATA AT FINISH (FINAL)
     OPEN(1,FILE='FINAL',STATUS='OLD',FORM='UNFORMATTED')
     READ(1) TIME
     READ(1) C
     READ(1) B
     IF (IPLOT.EQ.6) THEN
C  PLOT COARSE DATA, SKIP OVER FINE AND MEDIUM DATA IN FILE
      READ(1)ZCMAT
      HD=HEAD(8)
      WRITE(DEPTH,'(I2)')9
      WRITE(TIME8,'(F4.1)')TIME
      N1=7
      N2=8
      IMAT=3
      CALL AREA2D(4.1,7.2)
     ELSE
C  SELECT PROFILE TYPE FOR FINAL DATA PLOTTING
      WRITE(*,*)' SELECT OPTION:  (1) HORIZONTAL  (2) X-Z PROFILE'
      READ(*,*) IOPT
      IF (IOPT.EQ.2) THEN
13     WRITE(*,*)' AT WHAT Y LOCATION (1 TO 27)?'
       READ(*,*) J
       IF(J.LT.1.OR.J.GT.27) GOTO 13
      ENDIF
      IF(IPLOT.EQ.4) THEN
C  FINE FINAL DATA PLOT, HORIZONTAL SURFACE
       IF(IOPT.EQ.1) THEN
        WRITE(*,*)' AT WHAT DEPTH (1 TO 8)?'
        READ(*,*) K
        DO 14 I=1,27
         DO 14 J=1,27
14       ZMAT(I,J)=C(I,J,K)
        IMAT=1
        N1=5
        N2=8
        HD=HEAD(8)
        WRITE(DEPTH,'(I2)')K-1
```

```
            WRITE(TIME8,'(F4.1)')TIME
            CALL AREA2D(7,7)
           ELSE
C  FINE FINAL PLOT, XZ PROFILE
            DO 15 I=1,27
             DO 15 K=1,8
15           XZF(I,9-K)=C(I,J,K)
            IMAT=2
            N1=9
            N2=4
            HD=HEAD(9)
            WRITE(H9Y,'(I3)')J
            HEAD(9)=HD
            HD=HEAD(4)
            WRITE(TIME4,'(F4.1)')TIME
            CALL AREA2D(7.,2.4)
           ENDIF
          ELSE
C  MEDIUM FINAL PLOT, HORIZONTAL SURFACE
           IF(IOPT.EQ.1) THEN
            WRITE(*,*)' AT WHAT DEPTH (1 TO 10)?'
            READ(*,*) K
            DO 16 I=1,27
             DO 16 J=1,27
16           ZMAT(I,J)=B(I,J,K)
            IMAT=1
            N1=6
            N2=8
            HD=HEAD(8)
            WRITE(DEPTH,'(I2)')K*3-3
            WRITE(TIME8,'(F4.1)')TIME
            CALL AREA2D(7,7)
           ELSE
C  MEDIUM FINAL PLOT, XZ PROFILE
            DO 17 K=1,10
            DO 17 I=1,27
17          XZM(I,K)=B(I,J,11-K)
            IMAT=4
            N1=10
            N2=4
            HD=HEAD(10)
            WRITE(H10Y,'(I3)')J*3-2
            HEAD(10)=HD
            HD=HEAD(4)
            WRITE(TIME4,'(F4.1)')TIME
            CALL AREA2D(7.1,3)
           ENDIF
          ENDIF
         ENDIF
        ENDIF

C  ALLOW OPTION OF A 3D PLOT

        IF (IMAT.EQ.1) THEN
         WRITE(*,*)' DO YOU WISH A 3D PLOT?'
         READ(*,'(A1)') ANS
```

```
       IF (ANS.EQ.'Y') GOTO 100
      ENDIF

C  TEMPORARY FIX FOR CONTOUR PLOTTING
      I=10
      IF I=1
      CALL THKFRM(.01)

C  SET UP AND PLOT CONTOUR

      CALL BCOMON(20000)
      CALL XNAME('X-AXIS  DISTANCE IN MM',22)
      CALL HEADIN(HEAD(N1),N(N1),2,4)
      CALL HEADIN(HD,N(N2),2,4)
      CALL HEADIN(HEAD(11),47,2,4)
      CALL HEADIN(USERHD,100,2,4)
      IF (IMAT.EQ.1) THEN
       CALL YNAME('Y-AXIS  DISTANCE IN MM',22)
       CALL GRAF(0,3*SCALE,27*SCALE,0,3*SCALE,27*SCALE)
       CALL GRID(0,0)
       CALL CONMAK(ZMAT,27,27,T)
      ENDIF
      IF (IMAT.EQ.2) THEN
       CALL YNAME('Z-AXIS  DISTANCE IN MM',22)
       CALL GRAF(0,3,27,7,2,0)
       CALL GRID(0,0)
       CALL CONMAK(XZF,27,8,T)
      ENDIF
      IF (IMAT.EQ.3) THEN
       CALL YNAME('Y-AXIS  DISTANCE IN MM',22)
       CALL GRAF(0,27,180,0,27,315)
       CALL GRID(0,0)
       CALL CONMAK(ZCMAT,21,36,T)
      ENDIF
      IF (IMAT.EQ.4) THEN
       CALL YNAME('Z-AXIS  DISTANCE IN MM',22)
       CALL GRAF(0,9,81,27,3,0)
       CALL GRID(0,0)
       CALL CONMAK(XZM,27,10,T)
      ENDIF
      CALL CONANG(30)
      CALL CONMIN(4)
      CALL CONDIG(0)
      CALL CONTHN(.05)
      CALL CONLIN(0,'SOLID','LABELS',2,5)
      CALL CONLIN(1,'DOT','NOLABELS',1,1)
      CALL CONLIN(2,'DASH','NOLABELS',1,3)
      CALL CONLIN(3,'DOT','NOLABELS',1,1)
      CALL RASPLN(.25)
      CALL CONTUR(4,'LABELS','DRAW')
 99   CALL ENDPL(0)
      CLOSE(1)
999   WRITE(*,*) ' DO YOU WISH TO TRY ANOTHER CURVE?'
      READ(*,'(A1)')ANS
      IF (ANS.EQ.'Y') GOTO 1
      CALL DONEPL
```

```
      STOP

C  3D PLOT INSTEAD OF CONTOUR PLOT
  100 CALL HEADIN(HEAD(N1),N(N1),2,3)
      CALL HEADIN(HD,N(N2),2,3)
      CALL HEADIN(USERHD,100,2,3)
      CALL X3NAME('X-AXIS  DISTANCE IN MM',22)
      CALL Y3NAME('Y-AXIS  DISTANCE IN MM',22)
      CALL Z3NAME('TEMPERATURE IN DEGREES KELVIN$',100)
      CALL VOLM3D(10.,10.,8.)
      CALL VUABS(-15.,-15.,12.)
      CALL GRAF3D(0,3*SCALE,27*SCALE,0,3*SCALE,27*SCALE,300,T*4,
     *            T*24+300)
      CALL SURMAT(ZMAT,1,27,1,27,0)
      GOTO 99
C  FINAL FINE ZONE PROFILE PLOTTING
 1000 WRITE(*,'(/,1X,A)')' '
      WRITE(*,'(1X,A)')
     *'    THIS PROCEDURE ALLOWS PLACING MULTIPLE TEMPERATURE PROFILES',
     *'ONTO ONE PLOT.  THE INPUT FILES SHOULD BE NAMED AS FOLLOWS:',
     *'       FILE FILENAME A',
     *'WHERE FILENAME IS WHAT IS NORMALLY CALLED FILETYPE IN VMS.
ONCE',
     *'THE AXIS IS SPECIFIED FOR A PLOT IT MAY NOT BE CHANGED, THOUGH',
     *'THE OTHER TWO VARIABLES ARE FREE.  YOU WILL BE ASKED FOR A LABEL'
     *,'FOR EACH GRAPH.',' '
      DO 1006 I=1,27
 1006 XDATA(I)=I-.5
      YLABEL='TEMPERATURE IN DEGREES KELVIN$'
 1001 WRITE(*,*)' WHAT IS THE REFERENCE AXIS, X OR Y?'
      READ(*,'(A1)') AXS
      IF (AXS.NE.'Y'.AND.AXS.NE.'X') GOTO 1001
 1002 WRITE(*,*)' WHAT IS THE FILENAME FOR THIS CURVE?'
      READ(*,'(A8)')FNAME
      OPEN(1,FILE=FNAME,STATUS='OLD',FORM='UNFORMATTED')
      READ(1) TIME
      READ(1) C
      CLOSE(1)
      WRITE(*,*)' WHAT IS THE LABEL FOR THIS CURVE? (20 CHARACTERS MAX'
      WRITE(*,*)' AND MUST END IN A $)'
      READ(*,'(A20)') LGDTXT
      IF (AXS.EQ.'Y') THEN
       XLABEL='Y AXIS DISTANCE IN MM$'
       WRITE(*,*)
       WRITE(*,*)' WHAT ARE THE DESIRED X,Z COORDINATES? (1-27,1-8)?'
       READ (*,*) I,K
       DO 1003 J=1,27
1003   YDATA(J)=C(I,J,K)
      ELSE
       XLABEL='X AXIS DISTANCE IN MM$'
       WRITE(*,*)
       WRITE(*,*)' WHAT ARE THE DESIRED Y,Z COORDINATES? (1-27,1-8)?'
       READ (*,*) J,K
       DO 1004 I=1,27
1004   YDATA(I)=C(I,J,K)
      ENDIF
```

```
      WRITE(*,*)' DO YOU WISH ANOTHER PROFILE ON THIS PLOT?'
      READ(*,'(A1)')ANS
      IF(ANS.EQ.'Y') THEN
      CALL PLOTD(XDATA,YDATA,27,.FALSE.,'LINLIN',LGDTXT,HEAD(12),XLABEL,
     *           YLABEL)
      GOTO 1002
      ENDIF
      CALL PLOTD(XDATA,YDATA,27,.TRUE.,'LINLIN',LGDTXT,HEAD(12),XLABEL,
     *           YLABEL)
      GOTO 999
      END
```

B. ROSENTHAL VERIFICATION PROGRAM

This simple program uses the Rosenthal solution for a point source to predict the temperatures in lieu of the WELD program. It produces a FINAL file of similar format to the FINAL file of the weld program which can then be used to produce Final Fine or Medium plots of either the three dimensional or contour plots. In addition these results may be compared to the WELD program results by using the profile section of the PLOT program.

1. Source Listing of Program ROSEN

```
      PROGRAM ROSEN
C*************************************************************************
C  THIS PROGRAM SOLVES THE ROSENTHAL SOLUTION FOR A LOW POWER HEAT     *
C  SOURCE.  THIS IS TO VALIDATE A 3 DIMENSIONAL FINITE DIFFERENCE      *
C  MODEL OF WELDING ON A THICK PLATE.  THE OUTPUT FILE MIMICS A FINAL  *
C  FILE GENERATED BY THE FINITE DIFFERENCE MODEL FOR THE FINE AND      *
C  MEDIUM GRIDS AT TIME = 10 SECONDS.  THE FOLLOWING PARAMETERS WERE   *
C  USED:                                                               *
C  ARC VELOCITY = 4 MM/S   VOLTAGE = 30 VOLTS   CURRENT = 265 AMPS     *
C      EFFICIENCY = .005   QARC = 39.75 WATTS                          *
C  K = 53 W/M-K   C = 4.23 E6 J/M3-K   FINE GRID SPACING = .001 M      *
C  TO = 300 K                          MEDIUM GRID SPACING = .003 M    *
C                                                                      *
C BY ROBERT ULE   15 APR 1988                                          *
C*************************************************************************
      DIMENSION A(27,27,8),B(27,27,10)
      OPEN(1,FILE='ROSEN',STATUS='NEW',FORM='UNFORMATTED')
      WRITE(1)10.
      DO 1 I=1,27
       DO 1 J=1,27
       DO 1 K=1,8
        X=(I-14)*.001
        Y=(J-17)*.001
        Z=(K-1)*.001
        R=SQRT(X*X+Y*Y+Z*Z)
        IF (R.EQ.0.) THEN
         A(I,J,K)=1000.
        ELSE
        A(I,J,K)=300.+.1144*EXP(-160.*(Y+R)/R
        ENDIF
    1 CONTINUE
```

```
      WRITE(1)A
      DO 2 I=1,27
       DO 2 J=1,27
       DO 2 K=1,10
        X=(I-14)*.003
        Y=(J-15)*.003
        Z=(K-1)*.003
        R=SQRT(X*X+Y*Y+Z*Z)
        IF (R.EQ.0.) THEN
         B(I,J,K)=1000.
        ELSE
        B(I,J,K)=300.+.1144*EXP(-160.*(Y+R)/R
        ENDIF
    2 CONTINUE
      WRITE(1)B
      END
```

C. MISALIGNMENT PROGRAM

To study the result of tracking off of a weld seam, the main program WELD and the FIN subroutine were modified to simulate a seam. These modified versions are called WELDMA and FINMA, with the MA indicating misalignment. The WELDMA program differs in that it can position the arc in the x as well as the y direction. This modification is only in the portion of the code which adds the energy from the arc, and is included below for reference. The FINMA subroutine has been modified to include a new variable, MELT, which determines if the nodes adjacent to the seam has melted or not. If melting has occured, than MELT for that node is set equal to TRUE and the heat conduction across the seam is calculated normally. If the seam has not melted, than the heat conduction for the nodes adjacent to the seam are recalculated, using zero thermal conductivity. This modification is only made to the surface and the interior nodes, which is listed below.

1. Source Listing of WELDMA Modifications

```
      PROGRAM WELDMA
***  Modification of how arc energy is added  ***
C  POSITION HEAT SOURCE AND CALCULATE VOLUME WEIGHTING FACTOR   SUM'

       YARC=YVEL*TIME+73-NB*9-NC*3
       IF (TIME.LE.15.) THEN
        XARC=13.5
       ELSE
        XARC=13.5+XVEL*(TIME-15.)
       ENDIF
       SUM=0.
       DO 1 J=10,23
        DO 1 I=10,27
         SUM=SUM+EXP(-.10625*((J-YARC)**2+(I-XARC)**2))*.5
         DO 1 K=1,3
          SUM=SUM+EXP(-.10625*((J-YARC)**2+(I-XARC)**2+K**2))
```

```
    1  CONTINUE
       SUM=SUM/Q

C   ADD THE HEAT FROM THE ARC

       DO 2 J=10,23
        DO 2 I=10,27
         C(I,J,1)=C(I,J,1)+EXP(-.10625*((J-YARC)**2+(I-XARC)**2))/SUM
         DO 2 K=2,4
          C(I,J,K)=C(I,J,K)+EXP(-.10625*((J-YARC)**2+(I-XARC)**2+(K-1)
     *       **2))/SUM
    2  continue
```

2. Source Listing of FINMA Modifications

```
***  Modification to track status of the seam ***

C   THE INTERNAL NODES

       DO 1 I=2,26
        DO 1 J=2,26
         DO 1 K=2,7
          COUT(I,J,K)=FK(C(I,J,K))*FO(3)*(GK(C(I+1,J,K),C(I,J,K))+
     *    GK(C(I-1,J,K),C(I,J,K))+GK(C(I,J+1,K),C(I,J,K))+GK(C(I,J-1,K),
     *    C(I,J,K))+GK(C(I,J,K+1),C(I,J,K))+GK(C(I,J,K-1),C(I,J,K)))
    1 CONTINUE
C   RECALCULATE THE DIFFERENCE EQUATION FOR THE SEAM IF NOT MELTED
       DO 10 J=2,26
        DO 10 K=2,7
         IF ((C(13,J,K).GE.1773.).OR.(C(14,J,K).GE.1773.))
     *       MELT(J,K) = .TRUE.
         IF (MELT(J,K)) GOTO 10
         COUT(13,J,K)=FK(C(13,J,K))*FO(3)*(
     *   GK(C(12,J,K),C(13,J,K))+GK(C(13,J+1,K),C(13,J,K))+GK(C(13,J-1,K)
     *   ,C(13,J,K))+GK(C(13,J,K+1),C(13,J,K))+GK(C(13,J,K-1),C(13,J,K)))
         COUT(14,J,K)=FK(C(14,J,K))*FO(3)*(GK(C(15,J,K),C(14,J,K))+
     *   GK(C(14,J+1,K),C(14,J,K))+GK(C(14,J-1,K),
     *   C(14,J,K))+GK(C(14,J,K+1),C(14,J,K))+GK(C(14,J,K-1),C(14,J,K)))
   10 CONTINUE

C   THE TOP SURFACE

       TINF4=TINF**4
       DO 2 I=2,26
        DO 2 J=2,26
         COUT(I,J,1)=FK(C(I,J,1))*FO(3)*(GK(C(I+1,J,1),C(I,J,1))+
     *    GK(C(I-1,J,1),C(I,J,1))+GK(C(I,J+1,1),C(I,J,1))+
     *    GK(C(I,J-1,1),C(I,J,1))+2.*GK(C(I,J,2),C(I,J,1)))+
     *    BI(3)*(TINF4-C(I,J,1)**4)+BI(4)*(TINF-C(I,J,1))
    2 CONTINUE
C   RECALCULATE THE DIFFERENCE EQUATION FOR THE SEAM IF NOT MELTED
       DO 20 J=2,26
         IF ((C(13,J,1).GE.1773.).OR.(C(14,J,1).GE.1773.))
     *       MELT(J,1) = .TRUE.
         IF (MELT(J,1)) GOTO 20
         COUT(13,J,1)=FK(C(13,J,1))*FO(3)*(
     *    GK(C(12,J,1),C(13,J,1))+GK(C(13,J+1,1),C(13,J,1))+
```

```
     *    GK(C(13,J-1,1),C(13,J,1))+2.*GK(C(13,J,2),C(13,J,1)))+
     *    BI(3)*(TINF4-C(13,J,1)**4)+BI(4)*(TINF-C(13,J,1))
          COUT(14,J,1)=FK(C(14,J,1))*FO(3)*(GK(C(15,J,1),C(14,J,1))+
     *    GK(C(14,J+1,1),C(14,J,1))+
     *    GK(C(14,J-1,1),C(14,J,1))+2.*GK(C(14,J,2),C(14,J,1)))+
     *    BI(3)*(TINF4-C(14,J,1)**4)+BI(4)*(TINF-C(14,J,1))
   20 CONTINUE
```

D. LACK OF FUSION PROGRAMS

To study flaws (or regions of lack of fusion) the main program WELD and the FIN subroutine were modified to allow creating regions of zero thermal diffusion. In addition, an additional file, COMP, was created to produce the history of the temperatures behind the arc at 0.25 second intervals. This was one line of temperatures on the surface, at a y location of three millimeters directly behind the arc. The distance of three millimeters was insured since the arc was moving at one millimeter every .25 seconds and the grid spacing was one millimeter. When speeds other than four millimeters per second are used, this time should be modified accordingly. Thus after five seconds a total of 20 data sets had been taken. Only the modifications made to the program WELD and the subroutine FIN are listed below, since they were not extensive.

Two auxiliary programs were used to process the data from the file COMP. The first, called TABLE, merely converted the binary data in the file COMP to a readable data table. The second program, called FLAW, was used to fit the results of analyzing the data from the program TABLE. This was written in basic, since it was helpful for the program to be easily altered while trying to fit the data. The resulting fit from this program was reported in chapter 5.

When running the program, it was desired to perform multiple runs over one night. This was done by use of an exec. A sample exec is included below. This exec would copy the startup files from the backup disk B and then setup the input stack. The program then executes, and the exec prints out the desired results. The versions of plot listed are identical to the PLOT program but each does a predetermined set of plots. The modified startup files are discarded and the originals recopied to the A disk in preparation for running the next problem.

1. Source Listing of Program WELDLF Modifications

```
      PROGRAM WELDLF
C  THIS VERSION IS USED FOR INSERTING LACK OF FUSION ZONES IN THE
C  FINE ZONE.  THE OUTPUT FILES ARE MODIFIED IN NAME AND IT USES A
C  MODIFIED FINE ZONE SUBROUTINE FINLF.  THE SUBROUTINE FINLF IS WHERE
C  THE ACTUAL SIZE AND LOCATION OF THE LACK OF FUSION IS CONTROLLED.
C  THE POSITION OF THE LOF ZONE IS CONTROLLED WITH THE VARIABLE LOF.
```

```
* Added before first executable statement

      COMMON /INPUT/I1,I2,K1,K2,JL

* Added before entering the main program do loop

C  INPUT LACK OF FUSION ZONE DIMENSIONAL DATA

      READ(*,*) I1,I2,K1,K2,JL

C  INITIAL CONDITION VERIFICATION REPORT

      OPEN (11,FILE='VERIF',STATUS='NEW')
      WRITE (11,1234) TIME,OUT,Q,NB,NC,VEL,STEP,CSTEP,BSTEP,N,DELT,
     *  I1,I2,K1,K2,JL
      CLOSE(11)
      OPEN(11,FILE='COMP',STATUS='NEW',FORM='UNFORMATTED')
1234  FORMAT(' TIME = ',F6.2,/' OUT = ',F6.2,/' Q = ',E12.4,/
     +       ' NB = ',I3,/' NC = ',I3,/' VEL = ',F6.2,/' STEP = ',F6.2,/
     +     ' CSTEP = ',I3,/' BSTEP = ',I3,/' N = ',I4,/' DELT = ',F6.2,
     +     /' X = ',I3,' TO ',I3,/' Z = ',I3,' TO ',I3,/' Y = ',I3,
     +       ' LONG.')

* Typical modified Runge-Kutta step, calls subroutine FINLF versus FIN.
* All four of the calls to FIN are changed to FINLF

C  FIRST RUNGE-KUTTA ITERATION

      CALL COR(A,B,ASUM)
      CALL MED(A,B,CIN,BSUM)
      CALL FINLF(B,CIN,CSUM,LOF)

* Inserted immediately following completion of the Runge-Kutta sum.

C  OUTPUT COMPARISON OF ARC PROFILES

       IF (FLOAT(INT(N/50)).EQ.FLOAT(N)/50.) THEN
        WRITE(11) N,TIME,INT(YARC)-3,(T(C(III,INT(YARC)-3,1)),III=1,27)
       ENDIF
```

2. Source Listing of Subroutine FINLF Modifications

```
* The top section of the subroutine is shown, the rest is identical to
* the FIN subroutine.

      SUBROUTINE FINLF(B,C,COUT,LOF)
C  THIS SUBROUTINE HAS BEEN MODIFIED TO ALLOW INSERTION OF A LACK
C  OF FUSION ZONE.  THE ZONE MAY BE OF ANY SIZE BUT CANNOT INCLUDE
C  POINTS WITH I<3 OR I>25 OR J<3 OR J>25 OR K>6 SINCE THESE BOUNDARY
C  POINTS WOULD REQUIRE EXTENSIVE MODIFICATION OF THE SUBROUTINE.
C  THE CALLING PROGRAM, WELDLF, POSITIONS THE LOF ZONE ON THE FINE
C  GRID.  JL SPECIFIES THE LENGTH OF THE ZONE.  I1 AND I2 SPECIFY THE
C  WIDTH OF THE ZONE.  AND K1 AND K2 SPECIFY THE DEPTH OF THE ZONE.

      DIMENSION B(27,27,10),C(27,27,8),COUT(27,27,8),FO(3),BI(4)
      DIMENSION CK(27,27,8)
      COMMON NB,NC,YARC,SUM,TINF,BI,FO
```

```
      COMMON /INPUT/I1,I2,K1,K2,JL

      GK(T1,T2)=FK(T1)*(T1-T2)/(FK(T1)+FK(T2))

      DATA FKB/26.5/

C  CALCULATE THERMAL CONDUCTIVITY AT EACH NODE
      DO 10 I=1,27
       DO 10 J=1,27
        DO 10 K=1,8
         CK(I,J,K)=FK(C(I,J,K))
   10 CONTINUE

C  SET LACK OF FUSION ZONE TO ZERO CONDUCTIVITY

      DO 11 J=LOF-JL+1,LOF
       DO 11 I=I1,I2
        DO 11 K=K1,K2
  11    CK(I,J,K)=0.

*  A. THE INTERIOR NODES

      DO 1 I=2,26
       DO 1 J=2,26
        DO 1 K=2,7
         COUT(I,J,K)=FK(C(I,J,K))*FO(3)*(HK(I+1,J,K,I,J,K,C,CK)+
     *   HK(I-1,J,K,I,J,K,C,CK)+HK(I,J+1,K,I,J,K,C,CK)+HK(I,J-1,K,
     *   I,J,K,C,CK)+HK(I,J,K+1,I,J,K,C,CK)+HK(I,J,K-1,I,J,K,C,CK))
    1 CONTINUE

*  B. THE TOP SURFACE NODES

      TINF4=TINF**4
      DO 2 I=2,26
       DO 2 J=2,26
        COUT(I,J,1)=FK(C(I,J,1))*FO(3)*(HK(I+1,J,1,I,J,1,C,CK)+
     *   HK(I-1,J,1,I,J,1,C,CK)+HK(I,J+1,1,I,J,1,C,CK)+
     *   HK(I,J-1,1,I,J,1,C,CK)+2.*HK(I,J,2,I,J,1,C,CK))+
     *   BI(3)*(TINF4-C(I,J,1)**4)+BI(4)*(TINF-C(I,J,1))
    2 CONTINUE

* The function HK, which performs save function as GK but uses the
* precalculated thermal conductivities.

      REAL FUNCTION HK(I,J,K,L,M,N,C,CK)
      DIMENSION CK(27,27,8),C(27,27,8)
      A=CK(I,J,K)+CK(L,M,N)
      IF (A.EQ.0.) THEN
       HK=0.
      ELSE
       HK=CK(I,J,K)*(C(I,J,K)-C(L,M,N))/A
      ENDIF
      END
```

3. Source Listing of Program TABLE

```
      PROGRAM TABLE
      REAL T(20,27),TIME(20)
      INTEGER N(20),J(20)
      OPEN(1,FILE='COMP',FORM='UNFORMATTED')
      OPEN(2,FILE='PRINT',STATUS='NEW')
      DO 1 K=1,20
    1 READ(1) N(K),TIME(K),J(K),(T(K,I),I=1,27)
      WRITE(2,100) (N(K),TIME(K),J(K),K=1,20)
      WRITE(2,200) ((INT((T(K,I)-300)),K=1,20),I=1,27)
  100 FORMAT('  N    TIME   YARC-1',20(/I4,F8.2,I5))
  200 FORMAT(20I4)
      END
```

4. Source Listing of Program FLAW

```
10 'This program is used to process the data from running the program
20 'WELDLF.  After running WELDLF, the program TABLE is run to produce
30 'the history of temperature profiles behind the arc for different
40 'geometries of flaws.  This data is manually processed to determine
50 'the max temperature change from quasi-steadystate and the max
60 'percentage change from quasi-steadystate. This information, plus the
70 'flaw geometry and arc relative power is placed in a data file which
80 'can be read by this program in the order of: I1, I2, JL, K1, K2,
90 'Delta T max, Percent Delta T max, Relative Power.  The program then
100 'performs fits the data to a relationship between the max percent
105 'change in temperature from quasi-steadystate and the location of
110 'the flaw.  That relationship is:
120 '
130 '          T - Tqss        C*P*A
140 '         ----------   = ---------
150 '            Tqss          R*Z         2
160 '
170 'where:
180 '
190 '       X = (I1+I2)/2-14
200 '       A = (I2-I1)*JL
210 '       P = Relative Power
220 '       Z = K1-1
230 '       R = SQRT(X                    2 + Z 2)
240 '       C = Constant of the relationship
250 '
260 CLS
270 PRINT "          This Program determines the Relationship between"
280 PRINT "             Flaws and the Resultant Temperature Change."
290 PRINT
300 '  Input the Flaw parameters
310 INPUT "HOW MANY DATA POINTS ARE IN THE FILE";N
320 DIM X1(N),X2(N),DY(N),Z1(N),Z2(N),X(N),DX(N),A(N),R(N),DT(N),PT(N)
330 DIM P(N),Z(N),CT(N)
340 INPUT "What is the name of the file";FILE$
350 OPEN "I",#1,FILE$
360 FOR I=1 TO N
370 INPUT #1,X1(I),X2(I),DY(I),Z1(I),Z2(I),DT(I),PT(I),P(I)
380 NEXT I
390 '  Echo the input
```

```
400 PRINT
410 PRINT "No.";" " ;"X1","X2","DEL Y","Z1","Z2"
420 FOR I=1 TO N
430 PRINT I;" " ;X1(I),X2(I),DY(I),Z1(I),Z2(I)
440 NEXT I
450 '  Calculate the individual coefficient for each flaw.
460 PRINT : PRINT "No.","AREA","R","COR T"
470 FOR I=1 TO N
480 DX(I)=ABS(X2(I)-X1(I)) + 1
490 X(I)=(X1(I)+X2(I))/2 - 14
500 Z(I)=Z1(I)-1
510 R(I)=SQR(X(I)                          2+Z(I) 2)
520 A(I)=DX(I)*DY(I)
530 CT(I)=P(I)*A(I)/(R(I)*Z(I)               2)
540 PRINT I,A(I),R(I),CT(I)
550 NEXT I
560 INPUT "Strike Enter to continue";K$
570 '  Calculate the mean coefficient and display individual fits.
580 PRINT
590 PRINT "No.","TEMP","PERCENT","TEMP COR","PERCENT COR"
600 FOR I=1 TO N
610 PRINT I,DT(I),PT(I),DT(I)/CT(I),PT(I)/CT(I)
620 XI=PT(I)/CT(I)
630 SUM=SUM+XI
640 S2=XI*XI+S2
650 NEXT I
660 S=SQR((S2-SUM*SUM/N)/(N-1))
670 PRINT : PRINT "AVERAGE CORRELATION FIT = ";SUM/N,"STD DEV = ";S
680 END
```

5. Listing of a typical exec for repeated program execution

```
&TRACE
COPYFILE BASIC * B FILE = A
&STACK 11,14,4,4,3
EXEC RUN WELDLF
EXEC TDISK 15 DIS
EXEC DISSPLA PLOT1
EXEC SHERPA TEMPOUTP SHGRAPH T
EXEC DISSPLA PLOT2
EXEC SHERPA TEMPOUTP SHGRAPH T
EXEC DISSPLA PLOT3
EXEC SHERPA TEMPOUTP SHGRAPH T
EXEC PRINT FILE VERIF A
COPYFILE FILE COMP A = COMP9 B
EXEC DISSPLA PLOTC
EXEC SHERPA TEMPOUTP SHGRAPH T
EXEC RUN TABLE
EXEC PRINT FILE PRINT A
ERASE FILE * A
COPYFILE BASIC * B FILE = A
&STACK 11,14,4,4,2
EXEC RUN WELDLF
EXEC DISSPLA PLOT1
EXEC SHERPA TEMPOUTP SHGRAPH T
EXEC DISSPLA PLOT2
EXEC SHERPA TEMPOUTP SHGRAPH T
```

```
EXEC DISSPLA PLOT3
EXEC SHERPA TEMPOUTP SHGRAPH T
EXEC PRINT FILE VERIF A
COPYFILE FILE COMP A = COMP10 B
EXEC DISSPLA PLOTC
EXEC SHERPA TEMPOUTP SHGRAPH T
EXEC RUN TABLE
EXEC PRINT FILE PRINT A
ERASE FILE * A
```

E. COOLING RATE PROGRAMS

To measure the cooling rates during startup, steady state and shutdown the program WELD was modified and called WELDC. The modified subroutines are called FINC, MEDC and CORC. An additional subroutine RATES is added for calculating the cooling rates. Since the temperatures of interest are further from the arc, the Fine and Medium zones were extend to allow measuring these temperature. This modification approximately doubles the time to run the program. The other major modification was the way in which the heat is added to the material and how the molten region is modeled. The heat is only added to the surface nodes. To simulate the effect of arc pressure, a directional thermal conductivity is used, where the thermal conductivity in the Z direction is ten times that in the horizontal directions. This later effect only applies to those nodes which are above the melting temperature of the metal. This was implemented by an additional version of difference function GK, called HK, which is only used for differencing in the Z direction.

The RATES subroutine is called once every time step. It measures three separate cooling rates for the surface nodes in the heat affected zones. These cooling rates are determined by measuring the time it takes the temperature at a given absolute position to cool through a specified temperature range. This is performed in a three step process over all of the nodes currently in these fine and medium areas behind the arc. It first checks to see if the node temperature is now greater than the upper limit of the temperature band. If it is, this is noted for future reference with the logical array CT. The second check is to see if a node temperature is below the upper limit and previously above the upper band and yet never below it before. If this is the first time it has returned to below the upper band, this fact is noted for future reference in the logical array CT2. In addition, the time and temperature for this node are saved in the array CR. The third check is to see if a node temperature is below the lower limit of the temperature band having been previously above it. If it has returned to below the lower limit

for the first time then this is noted by setting the array CT for that node to false and the time and temperature for this node is again saved in the array CR.

This procedure is done in both the fine and medium zones. Due to the difference in sizes, only one out of nine fine nodal points will have a corresponding medium node. Thus if a nodal point temperature is not below the lower limit before the fine grid passes on, the program will not always report a cooling rate at the node. The subroutine RATES passes the arrays back to the main program WELDC which saves these files so that the program can be restarted. In addition, the CR array is separately outputted so that it can be processed to determine the cooling rates.

The program that determines the cooling rates is called RATEOUT. It reads the data file produced by WELDC and converts the data to cooling rates. It also provides the Y and R location of the node from the arc position at the mid-time of the cooling interval, as well as the absolute location of the node. If no cooling rate was determined for a location it will then provide some indication of the status of that location.

Due to the extensive modifications required to implement these changes, the new version of the source code is included in full. Extreme care was required to change all of the indices, it was almost as much work as writing the original code.

1. Source Listing of Program WELDC

```
      PROGRAM WELDC
************************************************************************************
*                                                                                  *
* DATE:   16 AUGUST 1988      WRITTEN BY:  ROBERT ULE                              *
* DATE:   17 AUGUST 1988      MODIFIED BY: ROBERT ULE                              *
*         USES DIRECTIONAL THERMAL CONDUCTIVITY IN THE WELD POOL AND               *
*         LONGER ARRAYS IN THE J DIRECTION TO STUDY COOLING RATES.                 *
*         THE ARC IS ADDED ONLY TO THE SURFACE NODES.                              *
*                                                                                  *
************************************************************************************
      DIMENSION A(21,36),AOUT(21,36),AIN(21,36),ASUM(21,36),
     *B(27,36,10),BOUT(27,36,10),BIN(27,36,10),BSUM(27,36,10),
     *C(27,45,8),COUT(27,45,8),CIN(27,45,8),CSUM(27,45,8),
     *FO(3),BI(4),CR(200,14,3,4),CT(200,14,3)
      INTEGER NDIV,I,J,K,M,BSTEP,CSTEP
      CHARACTER ANS
      LOGICAL YES,CT,CT2(200,14,3)
      COMMON NB,NC,YARC,SUM,TINF,BI,FO

      DATA A,B/10476*300.0/,C/9720*1.14237E8/,ASUM,BSUM/10476*0.0/
     *,AOUT,BOUT/10476*0.0/,CT/8400*.FALSE./,CT2/8400*
     *.FALSE./,CSUM/9720*0./,CR/33600*0./

C  SET TIME TO RUN THE PROBLEM AND FIXED CONDITIONS

      FINI = 20.
```

```
      NDIV=FINI*100
      XVEL=.0
      DELT=.01
      TINF=300.0

C  IF THIS IS A RESTART OF A PREVIOUS PROBLEM GOTO 100

C     GOTO 100

C  SET INITIAL VALUES FOR STARTING A PROBLEM

      YVEL=4.
      VOLT=30.
      AMP=100.
      EFF=.65

C  OPEN THE OUTPUT FILES

      OPEN(1,FILE='SURF',STATUS='NEW',FORM='UNFORMATTED')
      OPEN(2,FILE='FINAL',STATUS='NEW',FORM='UNFORMATTED')
      OPEN(3,FILE='CUT',STATUS='NEW',FORM='UNFORMATTED')
      OPEN(4,FILE='RATE',STATUS='NEW',FORM='UNFORMATTED')

C  THE INITIAL CONDITIONS

      TIME=0.
      STEP=3.
      OUT=.499
      N=0
      BSTEP=0
      CSTEP=0
      NB=3
      NC=10

C  WELD PARAMETERS

      QDOT=EFF*VOLT*AMP
      Q=QDOT*DELT*1.E9

C  REENTRY FOR RESTART

  200 CONTINUE

C  BOUNDARY CONDITION COEFFICIENTS

      FO(1)=DELT*.1636
      FO(2)=DELT*1.4722
      FO(3)=DELT*1000000.
      BI(1)=.001132
      BI(2)=.001132
      BI(3)=DELT*.00009299
      BI(4)=DELT*50000.

C  CALCULATE THE RUNGE-KUTTA APPROXIMATION
      DIS=TIME*YVEL
      YARC=YVEL*(TIME+DELT/2)+91-NB*9-NC*3
```

```
      DO 10 M=1,NDIV
       TIME=TIME+DELT
       DIS=TIME*YVEL
       N=N+1

C  POSITION HEAT SOURCE AND CALCULATE VOLUME WEIGHTING FACTOR   SUM'

       YARC=YVEL*(TIME+DELT/2)+91-NB*9-NC*3
       XARC=14.0
       SUM=0.
       DO 1 J=28,41
        DO 1 I=9,19
         SUM=SUM+EXP(-.10625*((J-YARC)**2+(I-XARC)**2))*.5
         DO 1 K=1,3
          SUM=SUM+EXP(-.10625*((J-YARC)**2+(I-XARC)**2+K**2))
    1  CONTINUE
       SUM=SUM/Q

C  ADD THE HEAT FROM THE ARC

       DO 2 J=28,41
        DO 2 I=9,19
         C(I,J,1)=C(I,J,1)+EXP(-.10625*((J-YARC)**2+(I-XARC)**2))/SUM
     *             *.5
        DO 2 K=2,4
         C(I,J,K)=C(I,J,K)+EXP(-.10625*((J-YARC)**2+(I-XARC)**2
     *           +(K-1)**2))/SUM
    2  CONTINUE

C  CONVERT ENTHALPY TO TEMPERATURE FOR THE FINE PLATE

      DO 60 I=1,27
       DO 60 J=1,45
        DO 60 K=1,8
         CIN(I,J,K)=T(C(I,J,K))
   60 CONTINUE

C  CALCULATE COOLING RATE DATA

      CALL RATES(B,CIN,CR,CT,CT2,NB,NC,TIME)

C  FIRST RUNGE-KUTTA ITERATION

      CALL CORC(A,B,ASUM)
      CALL MEDC(A,B,CIN,BSUM)
      CALL FINC(B,CIN,CSUM)

C  FIND T+K1/2

      DO 11 I=1,21
       DO 11 J=1,36
   11  AIN(I,J)=A(I,J)+ASUM(I,J)/2.
      DO 13 I=1,27
       DO 13 J=1,36
        DO 13 K=1,10
```

```
13    BIN(I,J,K)=B(I,J,K)+BSUM(I,J,K)/2.
      DO 12 I=1,27
       DO 12 J=1,45
        DO 12 K=1,8
        CIN(I,J,K)=T(C(I,J,K)+CSUM(I,J,K)/2.)
12    CONTINUE

C     SECOND ITERATION

      CALL CORC(AIN,BIN,AOUT)
      CALL MEDC(AIN,BIN,CIN,BOUT)
      CALL FINC(BIN,CIN,COUT)

C     FIND T+K2/2

      DO 21 I=1,21
       DO 21 J=1,36
       ASUM(I,J)=ASUM(I,J)+2.*AOUT(I,J)
21    AIN(I,J)=A(I,J)+AOUT(I,J)/2.
      DO 23 I=1,27
       DO 23 J=1,36
        DO 23 K=1,10
        BSUM(I,J,K)=BSUM(I,J,K)+2.*BOUT(I,J,K)
23    BIN(I,J,K)=B(I,J,K)+BOUT(I,J,K)/2.
      DO 22 I=1,27
       DO 22 J=1,45
        DO 22 K=1,8
        CSUM(I,J,K)=CSUM(I,J,K)+2.*COUT(I,J,K)
        CIN(I,J,K)=T(C(I,J,K)+COUT(I,J,K)/2.)
22    CONTINUE

C     THIRD ITERATION

      CALL CORC(AIN,BIN,AOUT)
      CALL MEDC(AIN,BIN,CIN,BOUT)
      CALL FINC(BIN,CIN,COUT)

C     FIND T+K3

      DO 31 I=1,21
       DO 31 J=1,36
       ASUM(I,J)=ASUM(I,J)+2.*AOUT(I,J)
31    AIN(I,J)=A(I,J)+AOUT(I,J)
      DO 33 I=1,27
       DO 33 J=1,36
        DO 33 K=1,10
        BSUM(I,J,K)=BSUM(I,J,K)+2.*BOUT(I,J,K)
33    BIN(I,J,K)=B(I,J,K)+BOUT(I,J,K)
      DO 32 I=1,27
       DO 32 J=1,45
        DO 32 K=1,8
        CSUM(I,J,K)=CSUM(I,J,K)+2.*COUT(I,J,K)
        CIN(I,J,K)=T(C(I,J,K)+COUT(I,J,K))
32    CONTINUE

C     FORTH ITERATION
```

```
      CALL CORC(AIN,BIN,AOUT)
      CALL MEDC(AIN,BIN,CIN,BOUT)
      CALL FINC(BIN,CIN,COUT)

C  PERFORM THE RUNGE-KUTTA SUM

      DO 40 I=1,21
       DO 40 J=1,36
40    A(I,J)=A(I,J)+(ASUM(I,J)+AOUT(I,J))/6
      DO 43 I=1,27
       DO 43 J=1,36
        DO 43 K=1,10
43     B(I,J,K)=B(I,J,K)+(BSUM(I,J,K)+BOUT(I,J,K))/6
      DO 41 I=1,27
       DO 41 J=1,45
        DO 41 K=1,8
41     C(I,J,K)=C(I,J,K)+(CSUM(I,J,K)+COUT(I,J,K))/6

C  STEP GRID IF REQUIRED

      IF (DIS.GE.STEP) THEN
       STEP=STEP+3.
        BSTEP=BSTEP+1

C  MOVE FINE GRID, FIRST AVERAGE FINE ENTHALP AND ASSIGN THE EQUIVALENT
C  TEMPERATURE TO THE MEDIUM GRID

       DO 51 IB=1,9
        HB1=0.
        HB2=0.
        HB3=0.
        DO 50 IC=IB*3-2,IB*3
         DO 50 JC=1,3
          HB1=C(IC,JC,1)+2.*C(IC,JC,2)+HB1
          DO 50 KC=3,5
           HB2=HB2+C(IC,JC,KC)
           HB3=HB3+C(IC,JC,KC+3)
50      CONTINUE
        B(IB+9,NC,1)=T(HB1/27)
        B(IB+9,NC,2)=T(HB2/27)
        B(IB+9,NC,3)=T(HB3/27)
51     CONTINUE

C  SHIFT FINE GRID

       DO 52 J=1,42
        JJ=J+3
        DO 52 K=1,8
         DO 52 I=1,27
52       C(I,J,K)=C(I,JJ,K)

C  ASSIGN MED TEMPS TO FINE GRID

       DO 53 I=10,18
        TB1=B(I,NC+15,1)
```

```
          TB2=B(I,NC+15,2)
          TB3=B(I,NC+15,3)
          DO 53 IC=(I-9)*3-2,(I-9)*3
           DO 53 JC=43,45
            C(IC,JC,1)=H(TB1)
            C(IC,JC,2)=H(TB1)
            DO 53 KC=3,5
             C(IC,JC,KC)=H(TB2)
             C(IC,JC,KC+3)=H(TB3)
   53    CONTINUE
         NC=NC+1

C  CHECK TO SEE IF TIME TO SHIFT MEDIUM GRID
         IF (BSTEP.EQ.3) THEN
          BSTEP=0

C  THEN AVERAGE MEDIUM GRID AND ASSIGN TO COARSE GRID

          DO 54 IA=1,9
           TA1=0.
           DO 55 IB=IA*3-2,IA*3
            DO 55 JB=1,3
             TA1=TA1+(B(IB,JB,1)+B(IB,JB,10))/2
             DO 55 KB=2,9
              TA1=TA1+B(IB,JB,KB)
   55      CONTINUE
           A(IA+6,NB)=TA1/81
   54     CONTINUE

C  SHIFT MED GRID

          DO 56 J=1,33
           JJ=J+3
           DO 56 I=1,27
            DO 56 K=1,10
   56       B(I,J,K)=B(I,JJ,K)

C  ASSIGN COARSE TEMPS TO MEDIUM GRID

          DO 57 I=7,15
           TA1=A(I,NB+12)
           DO 57 IB=(I-6)*3-2,(I-6)*3
            DO 57 JB=34,36
             DO 57 KB=1,10
              B(IB,JB,KB)=TA1
   57     CONTINUE
          NB=NB+1
          NC=10
         ENDIF
        ENDIF

C  OUTPUT RUNNING SOLUTION EVERY .5 SECONDS

        IF (TIME.GE.OUT) THEN
         OUT=OUT+.5
```

```
      WRITE(1) ((T(C(I,J,1)),I=1,27),J=1,45)
C  AVERAGE FINE ENTHALPIES AND ASSIGN EQUIVALENT TEMPERATURE TO THE
C  MEDIUM GRID
      DO 61 J=0,14
       DO 61 IB=1,9
        HB1=0.
        DO 66 IC=IB*3-2,IB*3
         DO 66 JC=J*3+1,J*3+3
66       HB1=C(IC,JC,1)+2*C(IC,JC,2)+HB1
61      B(9+IB,NC+J,1)=T(HB1/27.)
      WRITE(1)((B(I,J,1),I=1,27),J=1,36)
      WRITE(3)((T(C(I,INT(YARC),K)),I=1,27),K=1,8)
     ENDIF
10 CONTINUE

C  OUTPUT FINAL RESULTS

     WRITE(2) TIME
     WRITE(2)(((T(C(I,J,K)),I=1,27),J=1,45),K=1,8)
C  AVERAGE FINE ENTHALPIES AND ASSIGN EQUIVALENT TEMPERATURE TO THE
C  MEDIUM GRID
     DO 62 J=0,14
      DO 62 IB=1,9
       HB1=0.
       HB2=0.
       HB3=0.
       DO 63 IC=IB*3-2,IB*3
        DO 63 JC=1+J*3,3+J*3
         HB1=C(IC,JC,1)+2.*C(IC,JC,2)+HB1
         DO 63 KC=3,5
          HB2=HB2+C(IC,JC,KC)
          HB3=HB3+C(IC,JC,KC+3)
63    CONTINUE
      B(IB+9,NC+J,1)=T(HB1/27.)
      B(IB+9,NC+J,2)=T(HB2/27.)
      B(IB+9,NC+J,3)=T(HB3/27.)
62 CONTINUE
     WRITE(2)(((B(I,J,K),I=1,27),J=1,36),K=1,10)
C  AVERAGE MEDIUM TEMPERATURES AND ASSIGN TO COARSE GRID
     DO 64 J=0,11
      DO 64 IA=1,9
       TA1=0.
       DO 65 IB=IA*3-2,IA*3
        DO 65 JB=1+J*3,3+J*3
         TA1=TA1+(B(IB,JB,1)+B(IB,JB,10))/2
         DO 65 KB=2,9
          TA1=TA1+B(IB,JB,KB)
65    CONTINUE
      A(IA+6,NB+J)=TA1/81
64 CONTINUE
     WRITE(2)((A(I,J),I=1,21),J=1,36)

C  ALLOW OPTION OF RESTARTING THE PROBLEM

     INQUIRE(9,OPENED=YES)
     IF(YES) THEN
```

```
       REWIND(9)
      ELSE
       OPEN(9,FILE='RESTAR',STATUS='NEW',FORM='UNFORMATTED')
      ENDIF
      WRITE(9)OUT,Q,NB,NC,YVEL,STEP,CSTEP,BSTEP,N,DELT
      WRITE(9)A
      WRITE(9)B
      WRITE(9)C
      WRITE(4) CR,CT,CT2
      STOP

C  PROCEDURE FOR RESTARTING A PREVIOUS PROBLEM

  100 OPEN(2,FILE='FINAL',STATUS='OLD',FORM='UNFORMATTED')
      OPEN(4,FILE='RATE',STATUS='OLD',FORM='UNFORMATTED')

C  THE INITIAL CONDITIONS

      READ(2)TIME
      REWIND(2)
      IF (TIME.LT.(.499)) THEN
       OPEN(1,FILE='SURF',STATUS='NEW',FORM='UNFORMATTED')
       OPEN(3,FILE='CUT',STATUS='NEW',FORM='UNFORMATTED')
      ELSE
       OPEN(1,FILE='SURF',STATUS='OLD',FORM='UNFORMATTED')
       OPEN(3,FILE='CUT',STATUS='OLD',FORM='UNFORMATTED')
      ENDIF
      OPEN(9,FILE='RESTAR',STATUS='OLD',FORM='UNFORMATTED')
      READ(9)OUT,Q,NB,NC,YVEL,STEP,CSTEP,BSTEP,N,DELT

C  POSITION THE RUNNING OUTPUT FILES TO THE END OF THE FILES

      DO 102 L=1,N/50
       READ(1)((C(I,J,1),I=1,27),J=1,45)
       READ(1)((B(I,J,1),I=1,27),J=1,36)
       READ(3)((C(I,1,K),I=1,27),K=1,8)
  102 CONTINUE
      READ(4) CR,CT,CT2,DDELT
      REWIND(4)

C  INPUT THE TEMPERATURE/ENTHALPY MATRICES

      READ(9)A
      READ(9)B
      READ(9)C
      GOTO 200
      END
```

2. Source Listing of Subroutine FINC

```
      SUBROUTINE FINC(B,C,COUT)
C  THIS SUBROUTINE IS ASSOCIATED WITH THE PROGRAM WELDC, WHICH USES A
C  DIRECTIONAL THERMAL CONDUCTIVITY IN THE MOLTEN POOL.  IT HAS EXTENDED
C  ZONES IN THE J DIRECTION TO ALLOW BETTER RESOLUTION OF COOLING RATES.

      DIMENSION B(27,36,10),C(27,45,8),COUT(27,45,8),FO(3),BI(4)
      COMMON NB,NC,YARC,SUM,TINF,BI,FO
```

```
      GK(T1,T2)=FK(T1)*(T1-T2)/(FK(T1)+FK(T2))

      DATA FKB/26.5/

C  THE INTERNAL NODES

      DO 1 I=2,26
       DO 1 J=2,44
        DO 1 K=2,7
         COUT(I,J,K)=FK(C(I,J,K))*FO(3)*(GK(C(I+1,J,K),C(I,J,K))+
     *   GK(C(I-1,J,K),C(I,J,K))+GK(C(I,J+1,K),C(I,J,K))+GK(C(I,J-1,K),
     *   C(I,J,K))+HK(C(I,J,K+1),C(I,J,K))+HK(C(I,J,K-1),C(I,J,K)))
    1 CONTINUE

C  THE TOP SURFACE

      TINF4=TINF**4
      DO 2 I=2,26
       DO 2 J=2,44
        COUT(I,J,1)=FK(C(I,J,1))*FO(3)*(GK(C(I+1,J,1),C(I,J,1))+
     *   GK(C(I-1,J,1),C(I,J,1))+GK(C(I,J+1,1),C(I,J,1))+
     *   GK(C(I,J-1,1),C(I,J,1))+2.*HK(C(I,J,2),C(I,J,1)))+
     *   BI(3)*(TINF4-C(I,J,1)**4)+BI(4)*(TINF-C(I,J,1))
    2 CONTINUE

C  THE BOTTOM SURFACE (MEDIUM INTERFACE)

      DO 3 I=2,26
       IB=(I-.5)/3+10
       DO 3 J=2,44
        JB=(J-.5)/3+NC
        COUT(I,J,8)=FO(3)*(FK(C(I,J,8))*(GK(C(I+1,J,8),C(I,J,8))+
     *   GK(C(I-1,J,8),C(I,J,8))+GK(C(I,J+1,8),C(I,J,8))+GK(C(I,J-1,8),
     *   C(I,J,8))+HK(C(I,J,7),C(I,J,8)))+FKB*(B(IB,JB,4)-C(I,J,8)))
    3 CONTINUE

C  THE ENDS (MEDIUM INTERFACE)

      JB=NC-1
      JMAX=NC+15
      DO 4 I=2,27
       IB=(I-.5)/3+10
       DO 4 K=2,7
        KB=2
        IF(K.EQ.2) KB=1
        IF(K.GE.6) KB=3
        COUT(I,1,K)=FO(3)*(FK(C(I,1,K))*(GK(C(I+1,1,K),C(I,1,K))+
     *   GK(C(I-1,1,K),C(I,1,K))+GK(C(I,2,K),C(I,1,K))+GK(C(I,1,K+1),
     *   C(I,1,K))+GK(C(I,1,K-1),C(I,1,K)))+FKB*(B(IB,JB,KB)-
     *   C(I,1,K)))
        COUT(I,45,K)=FO(3)*(FK(C(I,45,K))*(GK(C(I+1,45,K),C(I,45,K))+
     *   GK(C(I-1,45,K),C(I,45,K))+GK(C(I,44,K),C(I,45,K))+
     *   HK(C(I,45,K+1),C(I,45,K))+HK(C(I,45,K-1),C(I,45,K)))+
     *   FKB*(B(IB,JMAX,KB)-C(I,45,K)))
    4 CONTINUE
```

```
C  THE SIDES (MEDIUM INTERFACE)

      DO 5 J=2,44
       JB=(J-.5)/3+NC
       DO 5 K=2,7
        KB=2
        IF(K.EQ.2) KB=1
        IF(K.GE.6) KB=3
         COUT(1,J,K)=FO(3)*(FK(C(1,J,K))*(GK(C(2,J,K),C(1,J,K))+
     *   GK(C(1,J+1,K),C(1,J,K))+GK(C(1,J-1,K),C(1,J,K))+
     *   HK(C(1,J,K+1),C(1,J,K))+HK(C(1,J,K-1),C(1,J,K)))+
     *   FKB*(B(9,JB,KB)-C(1,J,K)))
         COUT(27,J,K)=FO(3)*(FK(C(27,J,K))*(GK(C(26,J,K),C(27,J,K))+
     *   GK(C(27,J+1,K),C(27,J,K))+GK(C(27,J-1,K),C(27,J,K))+
     *   HK(C(27,J,K+1),C(27,J,K))+HK(C(27,J,K-1),C(27,J,K)))+
     *   FKB*(B(19,JB,KB)-C(27,J,K)))
    5 CONTINUE

C  THE END EDGES  (MEDIUM INTERFACE AND SURFACE EFFECTS ON TOP)

      KB=1
      JB=NC-1
      JMAX=NC+15
      DO 6 I=2,26
       IB=(I-.5)/3+10
       COUT(I,1,1)=FO(3)*(FK(C(I,1,1))*(GK(C(I+1,1,1),C(I,1,1))+
     *   GK(C(I-1,1,1),C(I,1,1))+GK(C(I,2,1),C(I,1,1))+
     *   2.*GK(C(I,1,2),C(I,1,1)))+FKB*(B(IB,JB,KB)-C(I,1,1)))+
     *   BI(3)*(TINF4-C(I,1,1)**4)+BI(4)*(TINF-C(I,1,1))
       COUT(I,45,1)=FO(3)*(FK(C(I,45,1))*(GK(C(I+1,45,1),C(I,45,1))+
     *   GK(C(I-1,45,1),C(I,45,1))+GK(C(I,44,1),C(I,45,1))+
     *   2.*GK(C(I,45,2),C(I,45,1)))+FKB*(B(IB,JMAX,KB)-C(I,45,1)))+
     *   BI(3)*(TINF4-C(I,45,1)**4)+BI(4)*(TINF-C(I,45,1))
       COUT(I,1,8)=FO(3)*(FK(C(I,1,8))*(GK(C(I+1,1,8),C(I,1,8))+
     *   GK(C(I-1,1,8),C(I,1,8))+GK(C(I,2,8),C(I,1,8))+
     *   GK(C(I,1,7),C(I,1,8)))+FKB*(B(IB,JB,3)+B(IB,NC,4)
     *   -2.*C(I,1,8)))
       COUT(I,45,8)=FO(3)*(FK(C(I,45,8))*(GK(C(I+1,45,8),C(I,45,8))+
     *   GK(C(I-1,45,8),C(I,45,8))+GK(C(I,44,8),C(I,45,8))+
     *   GK(C(I,45,7),C(I,45,8)))+FKB*(B(IB,JMAX,3)+B(IB,NC+14,4)-
     *   2.*C(I,45,8)))
    6 CONTINUE

C  THE SIDE EDGES (MEDIUM INTERFACE AND SURFACE EFFECTS ON TOP)

      DO 7 J=2,44
       JB=(J-.5)/3+NC
       COUT(1,J,1)=FO(3)*(FK(C(1,J,1))*(GK(C(1,J+1,1),C(1,J,1))+
     *   GK(C(1,J-1,1),C(1,J,1))+GK(C(2,J,1),C(1,J,1))+
     *   2.*GK(C(1,J,2),C(1,J,1)))+FKB*(B(9,JB,1)-C(1,J,1)))+
     *   BI(3)*(TINF4-C(1,J,1)**4)+BI(4)*(TINF-C(1,J,1))
       COUT(27,J,1)=FO(3)*(FK(C(27,J,1))*(GK(C(27,J+1,1),C(27,J,1))+
     *   GK(C(27,J-1,1),C(27,J,1))+GK(C(26,J,1),C(27,J,1))+
     *   2.*GK(C(27,J,2),C(27,J,1)))+FKB*(B(19,JB,1)-C(27,J,1)))+
     *   BI(3)*(TINF4-C(27,J,1)**4)+BI(4)*(TINF-C(27,J,1))
```

```
      COUT(1,J,8)=FO(3)*(FK(C(1,J,8))*(GK(C(1,J+1,8),C(1,J,8))+
     *   GK(C(1,J-1,8),C(1,J,8))+GK(C(2,J,8),C(1,J,8))+
     *   GK(C(1,J,7),C(1,J,8)))+FKB*(B(9,JB,3)+B(10,JB,4)
     *   -2.*C(1,J,8)))
      COUT(27,J,8)=FO(3)*(FK(C(27,J,8))*(GK(C(27,J+1,8),C(27,J,8))+
     *   GK(C(27,J-1,8),C(27,J,8))+GK(C(26,J,8),C(27,J,8))+
     *   GK(C(27,J,7),C(27,J,8)))+FKB*(B(19,JB,3)+B(18,JB,4)
     *   -2.*C(27,J,8)))
    7 CONTINUE

C  THE CORNER EDGES

      DO 8 K=2,7
       KB=2
       IF(K.EQ.2) KB=1
       IF(K.GE.6) KB=3
      COUT(1,1,K)=FO(3)*(FK(C(1,1,K))*(GK(C(2,1,K),C(1,1,K))+
     *   GK(C(1,2,K),C(1,1,K))+GK(C(1,1,K+1),C(1,1,K))+
     *   GK(C(1,1,K-1),C(1,1,K)))+FKB*(B(10,NC-1,KB)+B(9,NC,KB)-
     *   2.*C(1,1,K)))
      COUT(1,45,K)=FO(3)*(FK(C(1,45,K))*(GK(C(2,45,K),C(1,45,K))+
     *   GK(C(1,44,K),C(1,45,K))+GK(C(1,45,K+1),C(1,45,K))+
     *   GK(C(1,45,K-1),C(1,45,K)))+FKB*(B(10,NC+15,KB)+B(9,NC+14,KB)-
     *   2.*C(1,45,K)))
      COUT(27,1,K)=FO(3)*(FK(C(27,1,K))*(GK(C(26,1,K),C(27,1,K))+
     *   GK(C(27,2,K),C(27,1,K))+GK(C(27,1,K+1),C(27,1,K))+
     *   GK(C(27,1,K-1),C(27,1,K)))+FKB*(B(18,NC-1,KB)+B(19,NC,KB)-
     *   2.*C(27,1,K)))
      COUT(27,45,K)=FO(3)*(FK(C(27,45,K))*(GK(C(27,45,K),C(27,45,K))+
     *   GK(C(27,44,K),C(27,45,K))+GK(C(27,45,K+1),C(27,45,K))+
     *   GK(C(27,45,K-1),C(27,45,K)))+FKB*(B(19,NC+14,KB)+
     *   B(18,NC+15,KB)-2.*C(27,45,K)))
    8 CONTINUE

C THE CORNERS (MEDIUM INTERFACE AND TOP SURFACE EFFECTS)

      COUT(1,1,1)=FO(3)*(FK(C(1,1,1))*(GK(C(2,1,1),C(1,1,1))+
     *   GK(C(1,2,1),C(1,1,1))+GK(C(1,1,2),C(1,1,1)))+FKB*
     *   (B(10,NC-1,1)+B(9,NC,1)-2.*C(1,1,1)))+
     *   BI(3)*(TINF4-C(1,1,1)**4)+BI(4)*(TINF-C(1,1,1))
      COUT(1,45,1)=FO(3)*(FK(C(1,45,1))*(GK(C(2,45,1),C(1,45,1))+
     *   GK(C(1,44,1),C(1,45,1))+GK(C(1,45,2),C(1,45,1)))+FKB*
     *   (B(10,NC+15,1)+B(9,NC+14,1)-2.*C(1,45,1)))+
     *   BI(3)*(TINF4-C(1,45,1)**4)+BI(4)*(TINF-C(1,45,1))
      COUT(27,1,1)=FO(3)*(FK(C(27,1,1))*(GK(C(26,1,1),C(27,1,1))+
     *   GK(C(27,2,1),C(27,1,1))+GK(C(27,1,2),C(27,1,1)))+FKB*
     *   (B(18,NC-1,1)+B(19,NC,1)-2.*C(27,1,1)))+
     *   BI(3)*(TINF4-C(27,1,1)**4)+BI(4)*(TINF-C(27,1,1))
      COUT(27,45,1)=FO(3)*(FK(C(27,45,1))*(GK(C(26,45,1),C(27,45,1))+
     *   GK(C(27,44,1),C(27,45,1))+GK(C(27,45,2),C(27,45,1)))+FKB*
     *   (B(18,NC+15,1)+B(19,NC+14,1)-2.*C(27,45,1)))+
     *   BI(3)*(TINF4-C(27,45,1)**4)+BI(4)*(TINF-C(27,45,1))
      COUT(1,1,8)=FO(3)*(FK(C(1,1,8))*(GK(C(2,1,8),C(1,1,8))+
     *   GK(C(1,2,8),C(1,1,8))+GK(C(1,1,7),C(1,1,8)))+FKB*
     *   (B(10,NC-1,3)+B(9,NC,3)+B(10,NC,4)-3.*C(1,1,8)))
      COUT(1,45,8)=FO(3)*(FK(C(1,45,8))*(GK(C(2,45,8),C(1,45,8))+
```

```
     *   GK(C(1,44,8),C(1,45,8))+GK(C(1,45,7),C(1,45,8)))+FKB*
     *   (B(10,NC+15,3)+B(9,NC+14,3)+B(10,NC+14,4)-3.*C(1,45,8)))
      COUT(27,1,8)=FO(3)*(FK(C(27,1,8))*(GK(C(26,1,8),C(27,1,8))+
     *   GK(C(27,2,8),C(27,1,8))+GK(C(27,1,7),C(27,1,8)))+FKB*
     *   (B(18,NC-1,3)+B(19,NC,3)+B(18,NC,4)-3.*C(27,1,8)))
      COUT(27,45,8)=FO(3)*(FK(C(27,45,8))*(GK(C(26,45,8),C(27,45,8))+
     *   GK(C(27,44,8),C(27,45,8))+GK(C(27,45,7),C(27,45,8)))+FKB*
     *   (B(18,NC+15,3)+B(19,NC+14,3)+B(18,NC+14,4)-3.*C(27,45,8)))

      RETURN
      END

C  FUNCTION FOR FINDING THE EFFECT OF THERMAL CONDUCTIVITY WHEN DIRECT-
C     DIRECTIONAL IN THE WELD POOL.

      FUNCTION HK(T1,T2)
      A=FK(T1)
      B=FK(T2)
      C=1.
      IF (A.EQ.35.) A=175.
      IF (B.EQ.35.) THEN
       B=175.
       C=5.
      ENDIF
      HK=A*(T1-T2)*C/(A+B)
      END
```

3. Source Listing of Subroutine MEDC

```
      SUBROUTINE MEDC(A,B,C,BOUT)
C  THIS VERSION HAS BEEN MODIFIED TO HAVE LONGER ZONES IN THE J
C  DIRECTION TO ALLOW THE STUDY OF COOLING RATES.

      DIMENSION A(21,36),BOUT(27,36,10),B(27,36,10),BSUM(27,36,10),
     *C(27,45,8),FO(3),BI(4)
      COMMON NB,NC,YARC,SUM,TINF,BI,FO

*  A. THE INTERIOR NODES

      DO 7 I=2,26
       DO 7 J=2,35
        DO 777 K=2,4
         IF((I.GE.9.AND.I.LE.19.AND.J.GE.NC-1.AND.J.LE.NC+15)
     *   .AND..NOT.(K.EQ.4.AND.(I.EQ.9.OR.I.EQ.19.OR.J.EQ.NC-1.OR.J.EQ.
     *   NC+15)).AND..NOT.((J.EQ.NC-1.OR.J.EQ.NC+15).AND.(I.EQ.9.OR.I
     *   .EQ.19))) GOTO 777
         BOUT(I,J,K)=FO(2)*(B(I+1,J,K)+B(I-1,J,K)+B(I,J-1,K)+B(I,J+1,K)+
     *             B(I,J,K+1)+B(I,J,K-1)-6.*B(I,J,K))
777      CONTINUE
         DO 7 K=5,9
          BOUT(I,J,K)=FO(2)*(B(I+1,J,K)+B(I-1,J,K)+B(I,J-1,K)+B(I,J+1,K)+
     *             B(I,J,K+1)+B(I,J,K-1)-6.*B(I,J,K))
    7 CONTINUE

*  B. THE TOP AND BOTTOM SURFACE NODES

      DO 8 I=2,26
```

```
      DO 8 J=2,35
       BOUT(I,J,10)=FO(2)*(B(I+1,J,10)+B(I-1,J,10)+B(I,J+1,10)+
     *        B(I,J-1,10)+2*B(I,J,9)+BI(2)*TINF-(BI(2)+6.)*B(I,J,10))
        IF((I.GE.9.AND.I.LE.19.AND.J.GE.NC-1.AND.J.LE.NC+15)
     *   .AND..NOT.((J.EQ.NC-1.OR.J.EQ.NC+15).AND.(I.EQ.9.OR.I.EQ.
     *   19))) GOTO 8
       BOUT(I,J,1)=FO(2)*(B(I+1,J,1)+B(I-1,J,1)+B(I,J+1,1)+B(I,J-1,1)
     *            +2*B(I,J,2)+BI(2)*TINF-(BI(2)+6.)*B(I,J,1))
    8 CONTINUE

*   C. THE EXTERIOR END FACES (COARSE INTERFACE)

      DO 9 I=2,26
       IA=INT((I-.5)/3)+7
       DO 9 K=2,9
        BOUT(I,1,K)=FO(2)*(B(I+1,1,K)+B(I-1,1,K)+B(I,2,K)+B(I,1,K+1)+
     *          B(I,1,K-1)+.5*A(IA,NB-1)-5.5*B(I,1,K))
        BOUT(I,36,K)=FO(2)*(B(I+1,36,K)+B(I-1,36,K)+B(I,35,K)+
     *          B(I,36,K+1)+B(I,36,K-1)+.5*A(IA,NB+12)-5.5*B(I,36,K))
    9 CONTINUE

*   D. THE EXTERIOR SIDE FACES (COARSE INTERFACE)

      DO 10 J=2,35
       JA=INT((J-.5)/3)+NB
       DO 10 K=2,9
        BOUT(1,J,K)=FO(2)*(B(2,J,K)+B(1,J+1,K)+B(1,J-1,K)+B(1,J,K+1)+
     *          B(1,J,K-1)+.5*A(6,JA)-5.5*B(1,J,K))
        BOUT(27,J,K)=FO(2)*(B(26,J,K)+B(27,J+1,K)+B(27,J-1,K)+
     *          B(27,J,K+1)+B(27,J,K-1)+.5*A(16,JA)-5.5*B(27,J,K))
   10 CONTINUE

*   E. THE EXTERIOR END EDGES (COARSE INTERFACE)

      DO 11 I=2,26
       IA=INT((I-.5)/3)+7
       BOUT(I,1,1)=FO(2)*(B(I+1,1,1)+B(I-1,1,1)+B(I,2,1)+B(I,1,2)+
     *          .5*A(IA,NB-1)+BI(2)*TINF-(4.5+BI(2))*B(I,1,1))
       BOUT(I,1,10)=FO(2)*(B(I+1,1,10)+B(I-1,1,10)+B(I,2,10)+B(I,1,9)+
     *          .5*A(IA,NB-1)+BI(2)*TINF-(4.5+BI(2))*B(I,1,10))
       BOUT(I,36,1)=FO(2)*(B(I+1,36,1)+B(I-1,36,1)+B(I,35,1)+B(I,36,2)+
     *          .5*A(IA,NB+12)+BI(2)*TINF-(4.5+BI(2))*B(I,36,1))
       BOUT(I,36,10)=FO(2)*(B(I+1,36,10)+B(I-1,36,10)+B(I,35,10)+
     *     B(I,36,9)+.5*A(IA,NB+12)+BI(2)*TINF-(4.5+BI(2))*B(I,36,10))
   11 CONTINUE

*   F. THE COARSE SIDE EDGES (COARSE INTERFACE)

      DO 12 J=2,35
       JA=INT((J-.5)/3)+NB
       BOUT(1,J,1)=FO(2)*(B(2,J,1)+B(1,J+1,1)+B(1,J-1,1)+B(1,J,2)+
     *          .5*A(6,JA)+BI(2)*TINF-(4.5+BI(2))*B(1,J,1))
       BOUT(27,J,1)=FO(2)*(B(26,J,1)+B(27,J+1,1)+B(27,J-1,1)+B(27,J,2)+
     *          .5*A(16,JA)+BI(2)*TINF-(4.5+BI(2))*B(27,J,1))
       BOUT(1,J,10)=FO(2)*(B(2,J,10)+B(1,J+1,10)+B(1,J-1,10)+B(1,J,9)+
     *          .5*A(6,JA)+BI(2)*TINF-(4.5+BI(2))*B(1,J,10))
```

```
      BOUT(27,J,10)=FO(2)*(B(26,J,10)+B(27,J+1,10)+B(27,J-1,10)+
     *        B(27,J,9)+.5*A(16,JA)+BI(2)*TINF-(4.5+BI(2))*B(27,J,10))
   12 CONTINUE

*  G. THE EXTERIOR CORNER EDGES (COARSE INTERFACE)

      DO 13 K=2,9
       BOUT(1,1,K)=FO(2)*(B(2,1,K)+B(1,2,K)+B(1,1,K+1)+B(1,1,K-1)+
     *          .5*(A(6,NB)+A(7,NB-1))-5.*B(1,1,K))
       BOUT(27,1,K)=FO(2)*(B(26,1,K)+B(27,2,K)+B(27,1,K+1)+B(27,1,K-1)+
     *          .5*(A(16,NB)+A(15,NB-1))-5.*B(27,1,K))
       BOUT(1,36,K)=FO(2)*(B(2,36,K)+B(1,35,K)+B(1,36,K+1)+B(1,36,K-1)+
     *          .5*(A(6,NB+11)+A(7,NB+12))-5.*B(1,36,K))
       BOUT(27,36,K)=FO(2)*(B(26,36,K)+B(27,35,K)+B(27,36,K+1)+
     *          B(27,36,K-1)+.5*(A(16,NB+11)+A(15,NB+12))-5.*B(27,36,K))
   13 CONTINUE

*  H. THE EXTERIOR CORNERS (COARSE INTERFACE)

      BOUT(1,1,1)=FO(2)*(B(2,1,1)+B(1,2,1)+B(1,1,2)+.5*(A(6,NB)+
     *        A(7,NB-1))+BI(2)*TINF-(4.+BI(2))*B(1,1,1))

      BOUT(1,1,10)=FO(2)*(B(2,1,10)+B(1,2,10)+B(1,1,9)+.5*(A(6,NB)+
     *        A(7,NB-1))+BI(2)*TINF-(4.+BI(2))*B(1,1,10))
      BOUT(1,36,1)=FO(2)*(B(2,36,1)+B(1,35,1)+B(1,36,2)+.5*(A(6,NB+11)+
     *        A(7,NB+12))+BI(2)*TINF-(4.+BI(2))*B(1,36,1))
      BOUT(1,36,10)=FO(2)*(B(2,36,10)+B(1,35,10)+B(1,36,9)+.5*
     *        (A(6,NB+11)+A(7,NB+12))+BI(2)*TINF-(4.+BI(2))*B(1,36,10))
      BOUT(27,1,1)=FO(2)*(B(26,1,1)+B(27,2,1)+B(27,1,2)+.5*(A(16,NB)+
     *        A(15,NB-1))+BI(2)*TINF-(4.+BI(2))*B(27,1,1))
      BOUT(27,1,10)=FO(2)*(B(26,1,10)+B(27,2,10)+B(27,1,9)+.5*
     *        (A(16,NB)+A(15,NB-1))+BI(2)*TINF-(4.+BI(2))*B(27,1,10))
      BOUT(27,36,1)=FO(2)*(B(26,36,1)+B(27,35,1)+B(27,36,2)+.5*
     *      (A(16,NB+11)+A(15,NB+12))+BI(2)*TINF-(4.+BI(2))*B(27,36,1))
      BOUT(27,36,10)=FO(2)*(B(26,36,10)+B(27,35,10)+B(27,36,9)+.5*
     *      (A(16,NB+11)+A(15,NB+12))+BI(2)*TINF-(4.+BI(2))*B(27,36,10))

*  I. THE BOTTOM OF THE EXCLUSION ZONE (FINE INTERFACE)

      DO 20 I=10,18
       DO 20 J=NC,NC+14
        TC=0.
        DO 21 IC=3*(I-10)+1,3*(I-10)+3
         DO 21 JC=3*(J-NC)+1,3*(J-NC)+3
   21    TC=TC+C(IC,JC,8)
        BOUT(I,J,4)=FO(2)*(B(I+1,J,4)+B(I-1,J,4)+B(I,J+1,4)+B(I,J-1,4)
     *        +B(I,J,5)+TC/6.-6.5*B(I,J,4))
   20 CONTINUE

*  J.  THE SIDE OF THE EXCLUSION ZONE (FINE INTERFACE)

      DO 22 J=NC,NC+14
       DO 22 K=2,3
        TC1=0.
        TC2=0.
        DO 23 JC=3*(J-NC)+1,3*(J-NC)+3
```

```
      DO 23 KC=3*(K-2)+3,3*(K-2)+5
      TC1=TC1+C(1,JC,KC)
23    TC2=TC2+C(27,JC,KC)
     BOUT(9,J,K)=FO(2)*(B(8,J,K)+B(9,J+1,K)+B(9,J-1,K)+B(9,J,K+1)+
     *       B(9,J,K-1)+TC1/6.-6.5*B(9,J,K))
     BOUT(19,J,K)=FO(2)*(B(20,J,K)+B(19,J+1,K)+B(19,J-1,K)+
     *       B(19,J,K+1)+B(19,J,K-1)+TC2/6.-6.5*B(19,J,K))
   22 CONTINUE

*  K. THE ENDS OF THE EXCLUSION ZONE (FINE INTERFACE)

      DO 24 I=10,18
       DO 24 K=2,3
        TC1=0.
        TC2=0.
        DO 25 IC=3*(I-10)+1,3*(I-10)+3
         DO 25 KC=3*(K-2)+3,3*(K-2)+5
         TC1=TC1+C(IC,1,KC)
   25    TC2=TC2+C(IC,45,KC)
        BOUT(I,NC-1,K)=FO(2)*(B(I+1,NC-1,K)+B(I-1,NC-1,K)+B(I,NC-2,K)
     *     +B(I,NC-1,K+1)+B(I,NC-1,K-1)+TC1/6.-6.5*B(I,NC-1,K))
        BOUT(I,NC+15,K)=FO(2)*(B(I+1,NC+15,K)+B(I-1,NC+15,K)+B(I,NC+16,K
     *     )+B(I,NC+15,K+1)+B(I,NC+15,K-1)+TC2/6.-6.5*B(I,NC+15,K))
   24 CONTINUE

*  L.  THE SIDE EDGE OF THE EXCLUSION ZONE (FINE INTERFACE)

      DO 26 J=NC,NC+14
       TC1=0.
       TC2=0.
       DO 27 JC=3*(J-NC)+1,3*(J-NC)+3
        TC1=TC1+C(1,JC,1)+2*C(1,JC,2)
   27   TC2=TC2+C(27,JC,1)+2*C(27,JC,2)
       BOUT(9,J,1)=FO(2)*(B(8,J,1)+B(9,J+1,1)+B(9,J-1,1)+B(9,J,2)+
     *       TC1/6.+BI(2)*TINF-(5.5+BI(2))*B(9,J,1))
       BOUT(19,J,1)=FO(2)*(B(20,J,1)+B(19,J+1,1)+B(19,J-1,1)+B(19,J,2)+
     *       TC2/6.+BI(2)*TINF-(5.5+BI(2))*B(19,J,1))
   26 CONTINUE

*  M. THE END EDGE OF THE EXCLUSION ZONE (FINE INTERFACE)

      DO 28 I=10,18
       TC1=0.
       TC2=0.
       DO 29 IC=3*(I-10)+1,3*(I-10)+3
        TC1=TC1+C(IC,1,1)+2*C(IC,1,2)
   29   TC2=TC2+C(IC,45,1)+2*C(IC,45,2)
       BOUT(I,NC-1,1)=FO(2)*(B(I+1,NC-1,1)+B(I-1,NC-1,1)+B(I,NC-2,1)
     *     +B(I,NC-1,2)+TC1/6.+BI(2)*TINF-(5.5+BI(2))*B(I,NC-1,1))
       BOUT(I,NC+15,1)=FO(2)*(B(I+1,NC+15,1)+B(I-1,NC+15,1)+B(I,NC+16,1)
     *     +B(I,NC+15,2)+TC2/6.+BI(2)*TINF-(5.5+BI(2))*B(I,NC+15,1))
   28 CONTINUE

      RETURN
      END
```

4. Source Listing of Subroutine CORC

```
      SUBROUTINE CORC(A,B,AOUT)
C  THIS SUBROUTINE HAS BEEN MODIFIED TO USE EXTENDED ARRAYS IN THE J
C  DIRECTION TO ALLOW THE STUDY OF COOLING RATES.

      REAL A,B,AOUT
      DIMENSION A(21,36),AOUT(21,36),B(27,36,10),FO(3),BI(4)
      COMMON NB,NC,YARC,SUM,TINF,BI,FO

*  A. THE CORNERS

      AOUT(1,1)=0.
      AOUT(1,36)=0.
      AOUT(21,1)=0.
      AOUT(21,36)=0.

*  B. THE ENDS

      DO 1 I=2,20
       AOUT(I,1)=FO(1)*(A(I+1,1)+A(I-1,1)+A(I,2)+TINF*BI(1)-(3+BI(1))*
     *           A(I,1))
       AOUT(I,36)=FO(1)*(A(I+1,36)+A(I-1,36)+A(I,35)+TINF*BI(1)-(3+BI(1)
     *          )*A(I,36))
     1 CONTINUE

*  C. THE SIDES

      DO 2 J=2,35
       AOUT(1,J)=FO(1)*(A(1,J+1)+A(1,J-1)+A(2,J)+TINF*BI(1)-(3+BI(1))*
     *           A(1,J))
       AOUT(21,J)=FO(1)*(A(21,J+1)+A(21,J-1)+A(20,J)+TINF*BI(1)-(3+
     *           BI(1))*A(21,J))
     2 CONTINUE

*  D. THE MID POINTS

      NMIN=NB-1
      NMAX=NB+12
      DO 3 I=2,20
       DO 3 J=2,35
        IF (J.GE.NMIN.AND.J.LE.NMAX.AND.I.GE.7.AND.I.LE.15) GOTO 3
        IF (J.GE.NB.AND.J.LE.(NB+9).AND.(I.EQ.6.OR.I.EQ.16)) GOTO 3
        AOUT(I,J)=FO(1)*(A(I+1,J)+A(I-1,J)+A(I,J+1)+A(I,J-1)+BI(1)*TINF
     *            -(4+BI(1))*A(I,J))
     3 CONTINUE

*  E. THE MEDIUM ENDS TRANSITION

      DO 4 I=7,15
       TA=0.
       TB=0.
       DO 5 IB=(I-7)*3+1,(I-6)*3
        TA=B(IB,1,1)/2.+TA
        TA=B(IB,1,10)/2.+TA
        TB=B(IB,36,1)/2.+TB
```

```
      TB=B(IB,36,10)/2.+TB
      DO 5 K=2,9
       TA=B(IB,1,K)+TA
5      TB=B(IB,36,K)+TB
      AOUT(I,NMIN)=FO(1)*(A(I+1,NMIN)+A(I-1,NMIN)+A(I,NMIN-1)+TA/18.+
     *              BI(1)*TINF-(4.5+BI(1))*A(I,NMIN))
      AOUT(I,NMAX)=FO(1)*(A(I+1,NMAX)+A(I-1,NMAX)+A(I,NMAX+1)+TB/18.+
     *              BI(1)*TINF-(4.5+BI(1))*A(I,NMAX))
4     CONTINUE

*     F. THE MEDIUM SIDE TRANSITION

      DO 6 J=NB,NB+11
       TA=0.
       TB=0.
       DO 7 JB=(J-NB)*3+1,(J-NMIN)*3
        TA=B(1,JB,1)/2.+TA
        TA=B(1,JB,10)/2.+TA
        TB=B(27,JB,1)/2.+TB
        TB=B(27,JB,10)/2.+TB
        DO 7 K=2,9
         TA=B(1,JB,K)+TA
7       TB=B(27,JB,K)+TB
      AOUT(6,J)=FO(1)*(A(5,J)+A(6,J-1)+A(6,J+1)+TA/18.+
     *              BI(1)*TINF-(4.5+BI(1))*A(6,J))
      AOUT(16,J)=FO(1)*(A(17,J)+A(16,J-1)+A(16,J+1)+TB/18.+
     *              BI(1)*TINF-(4.5+BI(1))*A(16,J))
6     CONTINUE
      RETURN
      END
```

5. Source Listing of Subroutine RATES

```
      SUBROUTINE RATES(B,C,CR,CT,CT2,NB,NC,TIME)
*     THIS PROGRAM DETERMINES THE COOLING RATES WHILE RUNNING THE PROGRAM
*     WELDC.
      DIMENSION C(27,45,8),CR(200,14,3,4),CT(200,14,3),CT2(200,14,3)
     *,B(27,36,10)
      LOGICAL CT,CT2
      LOCAL=9*NB+3*NC-56
C     CHECK THE TEMPERATURES IN THE FINE ZONE
      DO 1 I=1,14
       J=0
       DO 1 JP=LOCAL,LOCAL+44
        J=J+1
C     CHECK TO SEE IF ABOVE THE UPPER LIMIT
        IF (C(I,J,1).GE.(825.0)) THEN
         CT(JP,I,1)=.TRUE.
        ENDIF
        IF (C(I,J,1).GE.(675.0)) THEN
         CT(JP,I,2)=.TRUE.
        ENDIF
        IF (C(I,J,1).GE.(1075.0)) THEN
         CT(JP,I,3)=.TRUE.
        ENDIF
C     CHECK TO SEE IF PREVIOUSLY ABOVE UPPER LIMIT BUT NOW BELOW.  IF SO
C     SAVE TIME AND TEMPERATURE.
```

```
      IF (C(I,J,1).LT.(825.).AND.CT(JP,I,1).AND.(.NOT.CT2(JP,I,1)))
     *  THEN
       CR(JP,I,1,1)=C(I,J,1)
       CR(JP,I,1,2)=TIME
       CT2(JP,I,1)=.TRUE.
      ENDIF
      IF (C(I,J,1).LT.(673.0).AND.CT(JP,I,2).AND.(.NOT.CT2(JP,I,2)))
     *  THEN
       CR(JP,I,2,1)=C(I,J,1)
       CR(JP,I,2,2)=TIME
       CT2(JP,I,2)=.TRUE.
      ENDIF
      IF (C(I,J,1).LT.(1073.0).AND.CT(JP,I,3).AND.(.NOT.CT2(JP,I,3)))
     *  THEN
       CR(JP,I,3,1)=C(I,J,1)
       CR(JP,I,3,2)=TIME
       CT2(JP,I,3)=.TRUE.
      ENDIF
C  IF TEMPERATURE IS LESS THAN THE LOWER LIMIT, SAVE THE TEMPERATURE AND
C  THE TIME.
      IF (CT2(JP,I,1).AND.(C(I,J,1).LT.795.).AND.CT(JP,I,1)) THEN
       CT(JP,I,1)=.FALSE.
       CR(JP,I,1,3)=C(I,J,1)
       CR(JP,I,1,4)=TIME
      ENDIF
      IF (CT2(JP,I,2).AND.(C(I,J,1).LT.423.).AND.CT(JP,I,2)) THEN
       CT(JP,I,2)=.FALSE.
       CR(JP,I,2,3)=C(I,J,1)
       CR(JP,I,2,4)=TIME
      ENDIF
      IF (CT2(JP,I,3).AND.(C(I,J,1).LT.773.).AND.CT(JP,I,3)) THEN
       CT(JP,I,3)=.FALSE.
       CR(JP,I,3,3)=C(I,J,1)
       CR(JP,I,3,4)=TIME
      ENDIF
    1 CONTINUE
C  CHECK THE TEMPERATURES IN THE MEDIUM ZONE
      DO 2 I=10,14
       IP=(I-9)*3-1
       DO 2 J=1,NC-1
        JP=9*NB+J*3-55
        IF (JP.LT.1) GOTO 2
C  CHECK TO SEE IF ABOVE THE UPPER LIMIT
        IF (B(I,J,1).GE.(825.0)) THEN
         CT(JP,IP,1)=.TRUE.
        ENDIF
        IF (B(I,J,1).GE.(675.0)) THEN
         CT(JP,IP,2)=.TRUE.
        ENDIF
        IF (B(I,J,1).GE.(1075.0)) THEN
         CT(JP,IP,3)=.TRUE.
        ENDIF
C  CHECK TO SEE IF PREVIOUSLY ABOVE UPPER LIMIT BUT NOW BELOW.  IF SO
C  SAVE TIME AND TEMPERATURE.
        IF (B(I,J,1).LT.(825.).AND.CT(JP,IP,1).AND.(.NOT.CT2(JP,IP,1)))
     *  THEN
```

```
          CR(JP,IP,1,1)=B(I,J,1)
          CR(JP,IP,1,2)=TIME
          CT2(JP,IP,1)=.TRUE.
         ENDIF
         IF (B(I,J,1).LT.(673.0).AND.CT(JP,IP,2).AND.(.NOT.CT2(JP,IP,2)))
     *   THEN
          CR(JP,IP,2,1)=B(I,J,1)
          CR(JP,IP,2,2)=TIME
          CT2(JP,IP,2)=.TRUE.
         ENDIF
         IF (B(I,J,1).LT.(1073.).AND.CT(JP,IP,3).AND.(.NOT.CT2(JP,IP,3)))
     *   THEN
          CR(JP,IP,3,1)=B(I,J,1)
          CR(JP,IP,3,2)=TIME
          CT2(JP,IP,3)=.TRUE.
         ENDIF
C   IF TEMPERATURE IS LESS THAN THE LOWER LIMIT, SAVE THE TEMPERATURE AND
C   THE TIME.
         IF (CT2(JP,IP,1).AND.(B(I,J,1).LT.795.).AND.CT(JP,IP,1)) THEN
          CT(JP,IP,1)=.FALSE.
          CR(JP,IP,1,3)=B(I,J,1)
          CR(JP,IP,1,4)=TIME
         ENDIF
         IF (CT2(JP,IP,2).AND.(B(I,J,1).LT.423.).AND.CT(JP,IP,2)) THEN
          CT(JP,IP,2)=.FALSE.
          CR(JP,IP,2,3)=B(I,J,1)
          CR(JP,IP,2,4)=TIME
         ENDIF
         IF (CT2(JP,IP,3).AND.(B(I,J,1).LT.773.).AND.CT(JP,IP,3)) THEN
          CT(JP,IP,3)=.FALSE.
          CR(JP,IP,3,3)=B(I,J,1)
          CR(JP,IP,3,4)=TIME
         ENDIF
       2 CONTINUE
         RETURN
         END
```

6. Source Listing of Program RATEOUT

```
         PROGRAM ROUT
C   USED TO OUTPUT COOLING RATES CALCULATED BY THE PROGRAM PLOTC
C   11-7-88 BY ROBERT ULE

         LOGICAL CT(200,14,3),CT2(200,14,3)
         REAL*4 CR(200,14,3,4),CRATE(200,14,3),RAD(200,14,3),Y(200,14,3)
         DATA CRATE/8400*0./,RAD/8400*0./,Y/8400*0./

         OPEN (1,FILE='RATE',STATUS='OLD',FORM='UNFORMATTED')
         OPEN (2,FILE='ROUT',STATUS='NEW')
         READ(1) CR,CT,CT2
         CLOSE (1)

C   CALCULATE COOLING RATE IF COOLED THROUGH THE TRANSITION ZONE
C   ELSE, INDICATE WHERE IT TEMPERATURE CYCLE THAT POINT IS.

         DO 10 J=1,200
          DO 10 I=1,14
```

```
        DO 10 K=1,3
         IF ((.NOT.CT(J,I,K)).AND.(.NOT.CT2(J,I,K))) CRATE(J,I,K)=-3.
         IF (CT(J,I,K).AND.(.NOT.CT2(J,I,K))) CRATE(J,I,K)=-2.
         IF (CT(J,I,K).AND.CT2(J,I,K)) CRATE(J,I,K)=-1.
         IF ((.NOT.CT(J,I,K)).AND.CT2(J,I,K)) THEN
          CRATE(J,I,K)=(CR(J,I,K,1)-CR(J,I,K,3))/(CR(J,I,K,4)
     *                -CR(J,I,K,2))
          YARC=34.+2.*(CR(J,I,K,2)+CR(J,I,K,4))
          Y(J,I,K)=YARC-J
          RAD(J,I,K)=SQRT(Y(J,I,K)**2+(14-I)**2)
         ENDIF
   10  CONTINUE

C   OUTPUT THE COOLING RATE DATA FOR THE THREE DIFFERENT CRITERIA AND
C   AT EACH POINT
       GOTO 60
       WRITE(2,'(1X,4(A,/))')
      *'COOLING RATE OUTPUT KEY:',
      *'-3 = STILL BELOW TRANSITION TEMPERATURE UPPER LIMIT',
      *'-2 = ABOVE TRANSITION TEMPERATURE UPPER LIMIT',
      *'-1 = COOLING BETWEEN UPPER AND LOWER TRANSITION LIMITS'
       DO 20 J=30,100
        WRITE(2,'(/A11,I3,/)')'Y POSITION ',J
        WRITE(2,'(A)') 'X POS     Y      R      MORRIS'
        DO 20 I=9,14
         WRITE(2,'(1X,I3,2X,2F8.2,F12.4)') I,
     *     Y(J,I,1),RAD(J,I,1),CRATE(J,I,1)
   20  CONTINUE
       DO 30 J=30,100
        WRITE(2,'(/A11,I3,/)')'Y POSITION ',J
        WRITE(2,'(A)') 'X POS     Y      R     MATERIAL'
        DO 30 I=8,14
         WRITE(2,'(1X,I3,2X,2F8.2,F12.4)') I,
     *     Y(J,I,2),RAD(J,I,2),CRATE(J,I,2)
   30  CONTINUE
       DO 40 J=30,100
        WRITE(2,'(/A11,I3,/)')'Y POSITION ',J
        WRITE(2,'(A)') 'X POS     Y      R     HYDROGEN'
        DO 40 I=8,14
         WRITE(2,'(1X,I3,2X,2F8.2,F12.4)') I,
     *     Y(J,I,3),RAD(J,I,3),CRATE(J,I,3)
   40  CONTINUE
   60  OPEN(10,FILE='PLOT1',STATUS='NEW')
       DO 70 J=30,96
       IF (CRATE(J,14,1).GT.0) WRITE(10,'(2F12.4)') FLOAT(J-36),
      * CRATE(J,14,1)
   70  CONTINUE
       OPEN(11,FILE='PLOT2',STATUS='NEW')
       DO 80 J=30,96
       IF (CRATE(J,13,1).GT.0) WRITE(11,'(2F12.4)') FLOAT(J-36),
      * CRATE(J,13,1)
   80  CONTINUE
       OPEN(12,FILE='PLOT3',STATUS='NEW')
       DO 90 J=30,96
       IF (CRATE(J,12,1).GT.0) WRITE(12,'(2F12.4)') FLOAT(J-36),
      * CRATE(J,12,1)
```

```
90  CONTINUE
    OPEN(13,FILE='PLOT4',STATUS='NEW')
    DO 95 J=30,96
    IF (CRATE(J,11,1).GT.0) WRITE(13,'(2F12.4)') FLOAT(J-36),
   * CRATE(J,11,1)
95  CONTINUE
    OPEN(14,FILE='PLOT5',STATUS='NEW')
    DO 96 J=30,96
    IF (CRATE(J,10,1).GT.0) WRITE(14,'(2F12.4)') FLOAT(J-36),
   * CRATE(J,10,1)
96  CONTINUE
    OPEN(15,FILE='PLOT6',STATUS='NEW')
    DO 97 J=30,96
    IF (CRATE(J,9,1).GT.0) WRITE(15,'(2F12.4)') FLOAT(J-36),
   * CRATE(J,9,1)
97  CONTINUE
    END
```

LIST OF REFERENCES

1. Goldak, J. and others, "Computational Heat Transfer for Weld Mechanics", *Advances in Welding Science and Technology,* Proceedings of an International Conference on Trends in Welding Research, pp. 15-20, 1986.

2. Mangonon, P. L. and Mahimkar, M. A., "A Three Dimensional Heat Transfer Finite Element Model of Submerged Arc Welding of HSLA Steels", *Advances in Welding Science and Technology,* Proceedings of an International Conference on Trends in Welding Research, pp. 33-45, 1986.

3. Smith, G.D., *Numerical Solution of Partial Differential Equations,* p. 122, Calaendon Press, 1985.

4. Lu, M. and Kou, S., "Power and Current Distributions in Gas Tungsten Arcs", *Welding Journal,* pp. 29s-34s, February 1988.

5. Goldak, J., Bibby, M., Moore, J., House, R. and Patel, B., "Computer Modeling of Heat Flow in Welds", *Metallurgical Transactions,* v. 17b, September 1986.

6. Van Der Houwen, P.J., *High Order Difference Schemes with Reduced Dispersion for Hyperbolic Differential Equations,* Centre for Mathematics and Computer Science, The Netherlands, 1985.

7. Doumanidis, C.C., Hale, M., and Hardt, D., "Multivariable Control of Arc Welding Processes", *Advances in Welding Science and Technology, Proceedings of an International Conference on Trends in Welding Research,* pp. 449-460, 1986.

8. Rosenthal, D., "The Theory of Moving Sources of Heat and its Application to Metal Treatments", *Transactions ASME,* v. 68, pp 849-869, November 1946.

9. Lancaster, J. E., *Metallurgy of Welding,* 4th ed., pp. 43-51, Allen & Unwin, 1987.

10. Thermal conductivity and heat capacities for the HY steels provided by William Morris, David Taylor Research Laboratory, Annapolis, Maryland. Source was United States Steel.

11. American Society for Testing and Materials Special Technical Publication 494, *Welding the HY Steels,"* by Flax, R. W., Keith, R. E., and Randall, M. D, April 1971.

12. Wray, P. J., "Mechanical, Physical and Thermal Data for Modeling the Solidification Processing of Steels", *Conference Proceedings, TMS-AIME,* pp. 245-257, 1981.

13. Begin, G., Boillot, J.P., and Michel, C., "Welding Adaptive Functions Performed Through Infrared Simplified Vision Schemes", *Optical Engineering,* September 1984.

14. Khan, M.A., Madsen, N.H., Goodling, J.S., and Chin, B.A.,"Infrared Thermography as a Control for the Welding Process", *Optical Engineering,* June 1986.

15. Morris, R. A., *Advanced Sensors for Automated Welding,* paper presented at the Arizona State University, 15 March 1988.

16. Khan, M.A., Madsen, N.H., Goodling, J.S., and Chin, B.A.,"Infrared Thermography as a Control for the Welding Process", *Optical Engineering,* June 1986.

17. Moore, S., *Projects in Progress,* p. 34, Journal of Metals, 1986.

18. *Oriel Light Sources, Monochromators and Detector Systems Catalogue,* v. 2, p.136, Oriel Company, 1988.

19. Button, K., *Infrared and Millimeter Waves,* v. 1,pp. 238-239, Academic Press, 1979.

20. Computer Associates International, *CA-DISSPLA Pocket Guide,* version 10.0, 1987.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	E. A. Metzbower Naval Research Laboratory Washington, D.C. 20375-5000	1
4.	Richard Morris David Taylor Research Laboratory Anapolis, MD 21402	2
5.	Department Chairman, Code 69 Department of Mechanical Engineering Naval Postgraduate School Monterey, CA 93943-5000	1
6.	Yogendra Joshi, Code 69JI Naval Postgraduate School Monterey, CA 93943-5000	2
7.	Alan Matteson David Taylor Research Laboratory Anapolis, MD 21402	1
8.	Robert Ule 27 Mustang Rd. Ranch Palos Verdes, CA 90274	2